

ITS
Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - TE 145561

SISTEM PENGHITUNGAN DEBIT AIR DAN PEMBAYARAN MENGUNAKAN SMART CARD PADA PDAM

Nur Atiqah Rianty Sari

NRP 2213030069

Rifqi Robuza Rohman

NRP 2213030075

Dosen Pembimbing

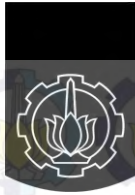
Ir. Hanny Boedinugroho, MT.

PROGRAM STUDI D3 TEKNIK ELEKTRO

Fakultas Teknologi Industri

Institut Teknologi Sepuluh Nopember

Surabaya 2016



ITS
Institut
Teknologi
Sepuluh Nopember

FINAL PROJECT - TE 145561

WATER FLOW CALCULATING SYSTEM AND PAYMENT USING SMART CARD ON PDAM

Nur Atiqah Rianty Sari

NRP 2213030069

Rifqi Robuza Rohman

NRP 2213030075

Advisor

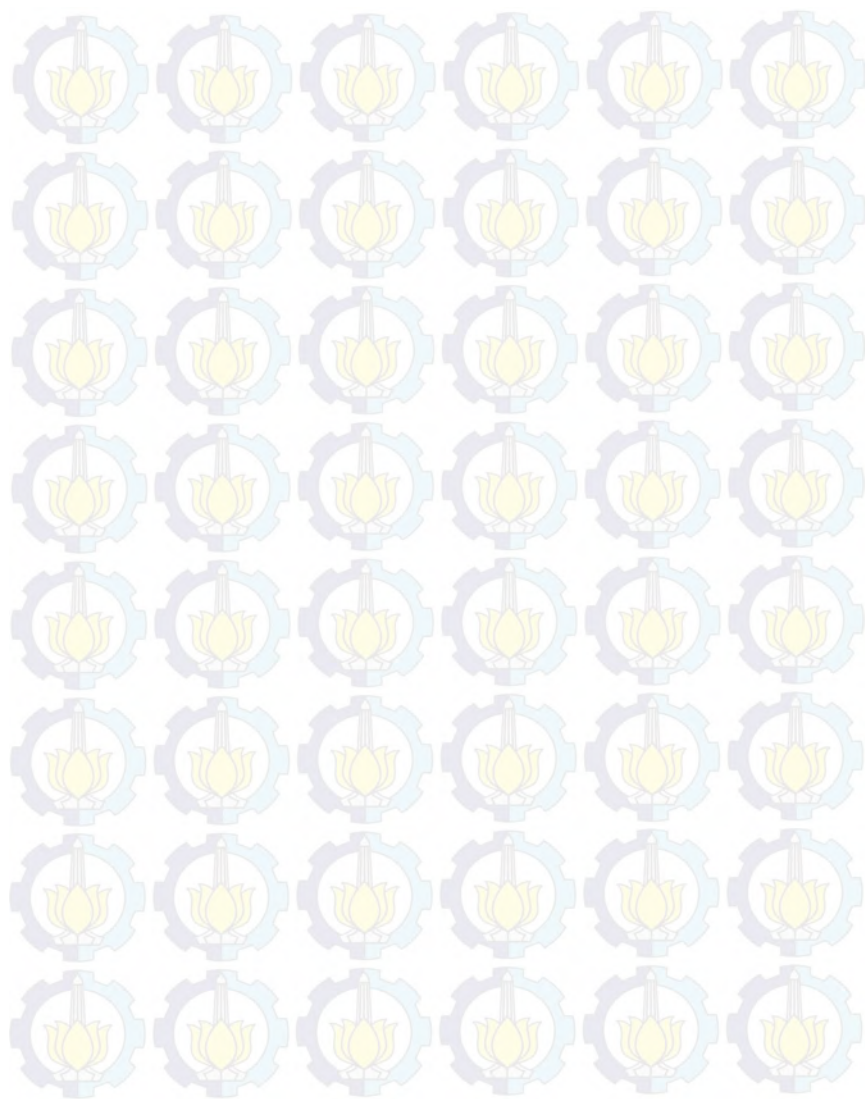
Ir. Hanny Boedinugroho, MT.

ELECTRICAL ENGINEERING D3 STUDY PROGRAM

Faculty of Industrial Technology

Institut Teknologi Sepuluh Nopember

Surabaya 2016



PERNYATAAN KEASLIAN TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul **“SISTEM PENGHITUNGAN DEBIT AIR DAN PEMBAYARAN MENGGUNAKAN SMART CARD PADA PDAM”** adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.


Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, 2 Juni 2016

Mahasiswa 1


Nur Atiqah Rianty S
NRP 2213030069

Mahasiswa 2


Rifqi Robuza Rohman
NRP 2213030075



**SISTEM PENGHITUNGAN DEBIT AIR DAN PEMBAYARAN
MENGUNAKAN *SMART CARD* PADA PDAM**


TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Ahli Madya Teknik
Pada

Bidang Studi Komputer Kontrol
Program Studi D3 Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui:

Dosen Pembimbing


Ir. Hanny Boedinugroho. MT.
NIP. 1961 07 06 1987 01 1001

**SURABAYA
JUNI, 2016**



**PERSETUJUAN PUBLIKASI KARYA ILMIAH
UNTUK KEPENTINGAN AKADEMIS**

Sebagai mahasiswa Institut Teknologi Sepuluh Nopember Surabaya, yang bertanda tangan di bawah ini saya :

Nama : NUR ATIOAH 25 / 250120012A 2
Nrp. : 224030069 / 221020071
Jurusan / Fak. : D3 TEKNIK ELEKTRO / FTI
Alamat kontak : Jl. GUBENG KERTAJAYA VI B NO 51
a. Email : risqirohulq@gmail.com
b. Telp/HIP : 002234002300

Menyatakan bahwa semua data yang saya *upload* di Digital Library ITS merupakan hasil final (revisi terakhir) dari karya ilmiah saya yang sudah diuvaluasi oleh dosen pengaji. Apabila dikemudian hari ditemukan ada ketidaksesuaian dengan kenyataan, maka saya bersedia menerima sanksi.

Demi perkembangan ilmu pengetahuan, saya menyetujui untuk memberikan Hak Bebas Royalti Non-Eksklusif (*Non-Exclusive Royalti-Free Right*) kepada Institut Teknologi Sepuluh Nopember Surabaya atas karya ilmiah saya yang berjudul :

SISTEM PENGHITUNGAN DEBIT NIR DAN PEMBAHARAN MENGGUNAKAN
SMART CARD PADA PDAM

Dengan Hak Bebas Royalti Non-Eksklusif ini, Institut Teknologi Sepuluh Nopember Surabaya berhak menyimpan, mengalih-media/format-kan, mengelolanya dalam bentuk pangkalan data (*database*), mendistribusikannya, dan menampilkan/mempublikasikannya di internet atau media lain untuk kepentingan akademis tanpa meminta ijin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta. Saya bersedia menanggung secara pribadi, segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta dalam karya ilmiah saya ini tanpa melibatkan pihak Institut Teknologi Sepuluh Nopember Surabaya.

Demikian pernyataan ini saya buat dengan sebenarnya.



Dosen Pembimbing 1

Ir. Harani Pradinawati, MT.

NIP. 1961 07 06 1947 01 1001

Dibuat di : Surabaya

Pada tanggal : 27 Juli 2015

Yang menyatakan,

NISOL 250120012A 2

Nrp. 224 030 069

KETERANGAN :

Tanda tangan pembimbing wajib dibubuhi stempel jurusan.



SISTEM PENGHITUNGAN DEBIT AIR DAN PEMBAYARAN MENGUNAKAN SMART CARD PADA PDAM

Nama : Nur Atiqah Rianty Sari
Nama : Rifqi Robuza Rohman
Pembimbing : Ir. Hanny Boedinugroho, MT.

ABSTRAK

Masyarakat masih sangat di rugikan akibat pembayaran tagihan Perusahaan Daerah Air Minum (PDAM) yang setiap bulannya masih menggunakan cara manual yaitu dengan melakukan pembayaran ke loket. Masalah yang sering terjadi adalah pembayaran yang dilakukan tidak sesuai dengan meteran pemakaian yang digunakan. Ada kalanya apabila petugas Perusahaan Daerah Air Minum (PDAM) tidak melakukan pengecekan meteran air, maka pelanggan akan dikenakan biaya pajak saja. Tetapi pada bulan selanjutnya, pelanggan akan membayar biaya yang lebih besar.

Oleh karena itu, akan dirancang pembayaran menggunakan *smart card* untuk PDAM yang dilatar belakangi masalah tersebut. Rancangan alat ini menggunakan sensor *water flow meter* yang memiliki kegunaan untuk mengukur jumlah debit air yang mengalir dalam pipa. Di dalam *water flow meter* ini terdapat suatu rotor yang akan berputar apabila ada aliran air. Jadi, saat pembayaran air menggunakan *smart card* yang digunakan sudah habis, akan ada suatu *solenoid valve* yang menutup agar air tidak dapat mengalir.

Hasil akhir dari Tugas Akhir ini adalah suatu alat yang dapat menghitung debit air dengan persentase *error* maksimum dari sensor *water flow meter* sebesar 6% dan pembayaran PDAM menggunakan *smart card* dengan persentase *error* sebesar 0%.

Kata Kunci : *Water Flow Meter, Smart Card, Solenoid Valve*



WATER FLOW CALCULATING SYSTEM AND PAYMENT USING SMART CARD ON PDAM

Name : Nur Atiqah Rianty Sari
Name : Rifqi Robuza Rohman
Advisor : Ir. Hanny Boedinugroho, MT.

ABSTRACT

The society still disadvantaged with the monthly payment of a Local Water Company (PDAM) if it still use the manual way to make a payment to the counter. A problem that often occurs is the payment does not match with the meter. There are times when the officer of the Local Water Company (PDAM) doesn't checking the monthly meter, then the costumers will charge tax only. But in the next month, the society will pay a larger fee.

Therefore, the payment of PDAM will be designed using the smart card which will be based on the problem. The design of the device using water flow meter sensor to measure the amount of water flowing in the discharge pipe. In the water flow meter sensor, there is a rotor that will spin when water are flow. So, when using smart card for payment of PDAM already empty. There will be closed solenoid valve, and the water cannot flow.

The result of this final project is a device that use to calculate the water discharge with maximum error percentage of water flow meter sensor is 6% and the payments methods using smart cards, the percentage error is 0%.

Keywords : *Water Flow Meter, Hall Effect, Smart Card, Solenoid Valve*



KATA PENGANTAR

Puji syukur penulis panjatkan kehadiran Allah SWT yang selalu memberikan rahmat dan hidayah-Nya sehingga Tugas Akhir ini dapat terselesaikan dengan baik. Shalawat serta salam semoga selalu dilimpahkan kepada Rasulullah Muhammad SAW, keluarga, dan semua pihak yang mendukung terselesaikan Tugas Akhir ini.

Tugas Akhir ini disusun untuk memenuhi sebagian persyaratan guna menyelesaikan pendidikan Diploma pada Bidang Studi Komputer Kontrol, Jurusan D3 Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember Surabaya dengan judul:

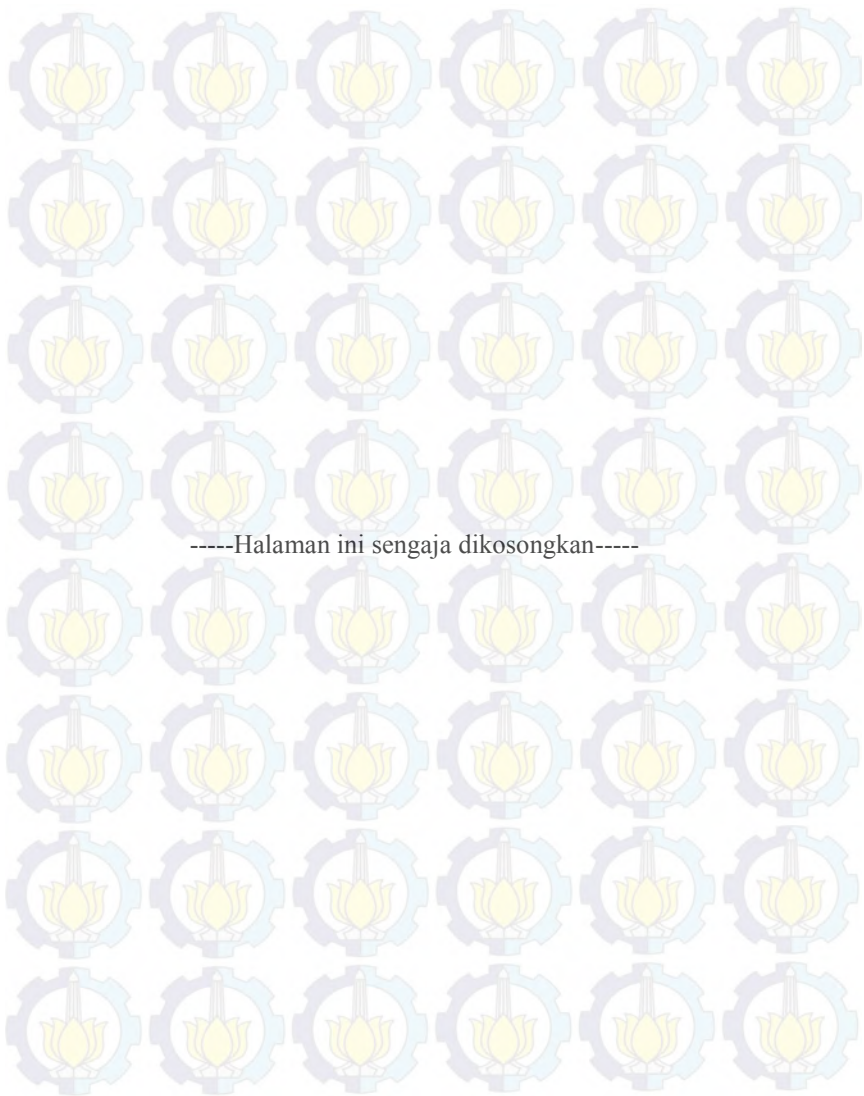
SISTEM PENGHITUNGAN DEBIT AIR DAN PEMBAYARAN MENGGUNAKAN SMART CARD PADA PDAM

Penulis mengucapkan terima kasih kepada Ibu dan Bapak penulis yang memberikan berbagai bentuk doa serta dukungan tulus tiada henti, Bapak Ir. Hanny Boedinoegroho, MT atas segala bimbingan ilmu, moral, dan spiritual dari awal hingga terselesaikannya Tugas Akhir ini. Penulis juga mengucapkan banyak terima kasih kepada semua pihak yang telah membantu baik secara langsung maupun tidak langsung dalam proses penyelesaian Tugas Akhir ini.

Penulis menyadari dan memohon maaf atas segala kekurangan pada Tugas Akhir ini. Akhir kata, semoga Tugas Akhir ini dapat bermanfaat dalam pengembangan keilmuan di kemudian hari.

Surabaya, 2 Juni 2016

Penulis



DAFTAR ISI

HALAMAN JUDUL.....	i
HALAMAN JUDUL.....	iii
PERNYATAAN KEASLIAN TUGAS AKHIR	v
HALAMAN PENGESAHAN	vii
ABSTRAK	xi
<i>ABSTRACT</i>	xiii
KATA PENGANTAR.....	xv
DAFTAR ISI.....	xvii
DAFTAR GAMBAR	xix
DAFTAR TABEL	xxi
BAB I PENDAHULUAN	Error! Bookmark not defined.
1.1 Latar Belakang	Error! Bookmark not defined.
1.2 Permasalahan	Error! Bookmark not defined.
1.3 Batasan Masalah.....	Error! Bookmark not defined.
1.4 Tujuan	Error! Bookmark not defined.
1.5 Metodologi Penelitian	Error! Bookmark not defined.
1.6 Sistematika Laporan.....	Error! Bookmark not defined.
1.7 Relevansi.....	Error! Bookmark not defined.
BAB II TEORI DASAR	Error! Bookmark not defined.
2.1 Mikrokontroler ATmega328	Error! Bookmark not defined.
2.2 RFID Mifare RC522.....	Error! Bookmark not defined.
2.3 RFID Tag.....	7
2.4 <i>Liquid Cristal Display</i> 20x4 (LCD).....	Error! Bookmark not defined.
2.5 <i>Solenoid Valve</i>	Error! Bookmark not defined.
2.6 <i>Buzzer</i>	Error! Bookmark not defined.
2.7 <i>Water Flow Meter</i>	Error! Bookmark not defined.
2.8 Pemrograman Arduino IDE.....	Error! Bookmark not defined.
2.9 <i>Relay</i>	Error! Bookmark not defined.
2.10 <i>Power Supply</i> 12V	Error! Bookmark not defined.
BAB III PERANCANGAN DAN PEMBUATAN SISTEM KONTROL	Error! Bookmark not defined.
3.1 Perancangan dan Pembuatan Mekanik.....	Error! Bookmark not defined.
3.2 Perancangan dan Pembuatan Perangkat Elektronik.....	Error! Bookmark not defined.

3.2.1 Rangkaian Power Supply	Error! Bookmark not defined.
3.2.2 Rangkaian <i>Minimum System</i> ATmega328	Error! Bookmark not defined.
3.2.3 Rangkaian LCD	Error! Bookmark not defined.
3.2.4 Rangkaian <i>Driver Relay</i>	Error! Bookmark not defined.
3.3 Perancangan dan Pembuatan Perangkat Lunak (<i>Software</i>)	Error! Bookmark not defined.
3.3.1 Pembuatan <i>FlowChart</i>	Error! Bookmark not defined.
3.3.2 Perancangan Program pada Arduino	Error! Bookmark not defined.
3.3.3 Perancangan Program <i>Smartcard</i>	Error! Bookmark not defined.
3.3.4 Perancangan Program <i>Flow Meter</i>	Error! Bookmark not defined.
BAB IV HASIL DAN IMPLEMENTASI	Error! Bookmark not defined.
4.1 Pengujian Rangkaian Power Supply	Error! Bookmark not defined.
4.2 Pengujian <i>Flow Meter</i> Sensor	Error! Bookmark not defined.
4.3 Pengujian <i>Smart Card</i>	Error! Bookmark not defined.
DAFTAR PUSTAKA	3
LAMPIRAN A	53
LAMPIRAN B	75
DAFTAR RIWAYAT HIDUP	77

DAFTAR GAMBAR

Gambar 2.1	ATmega328	5
Gambar 2.2	RFID Mifare RC522	7
Gambar 2.3	RFID <i>Tag</i>	7
Gambar 2.4	Struktur Data pada Mifare Card.....	8
Gambar 2.5	LCD 20X4	10
Gambar 2.6	Solenoid Valve.....	11
Gambar 2.7	Buzzer.....	12
Gambar 2.8	<i>Water Flow Meter Sensor</i>	13
Gambar 2.9	Tampilan Arduino IDE 1.6.1	14
Gambar 2.10	<i>Relay</i>	15
Gambar 2.11	Diagram Blok DC <i>Power Supply</i>	16
Gambar 2.12	Tabel Tarif Air pada PDAM	17
Gambar 3.1	Diagram fungsional Pembuatan Perangkat Elektronik ...	19
Gambar 3.2	Perancangan Mekanik Tampak Atas.....	20
Gambar 3.3	Perancangan Mekanik Tampak Samping.....	20
Gambar 3.4	Perancangan Mekanik Tampak Dalam	21
Gambar 3.5	Perancangan Mekanik Keseluruhan.....	21
Gambar 3.6	Rangkaian <i>Power Supply</i>	23
Gambar 3.7	<i>Board Minimum System</i> ATmega328	23
Gambar 3.8	Rangkaian <i>Minimum System</i> ATmega328	24
Gambar 3.9	Rangkaian LCD	24
Gambar 3.10	Skema Rangkaian <i>Relay</i>	25
Gambar 3.11	<i>Flowchart System</i>	26
Gambar 3.12	Tampilan <i>Software</i> Arduino.....	28
Gambar 3.13	Tampilan Arduino Ketika Tidak Terjadi <i>Error</i> pada Saat <i>Verify</i> Program	29
Gambar 3.14	Memilih <i>Board</i> Arduino	30
Gambar 3.15	Memilih <i>Port</i> pada Arduino	30
Gambar 3.16	Menambah <i>Library</i> MFRC522 pada Arduino.....	31
Gambar 3.17	Menambah <i>Library</i> pada Arduino.....	31
Gambar 3.18	Menginisialisasi Pin pada Arduino	32
Gambar 3.19	Program <i>Void Setup</i> pada Arduino.....	32
Gambar 3.20	<i>Void Setup</i> yang Kosong pada Arduino	32
Gambar 3.21	Inisialisasi Pin pada <i>Board</i> Arduino	33
Gambar 3.22	Pemrograman Fungsi <i>Setup</i> pada Arduino.....	33

Gambar 3.23	Memilih Salah Satu <i>Byte</i> Data dari Sebuah Blok pada <i>Smart Card</i>	34
Gambar 3.24	Program untuk Membaca <i>Smart Card</i> Pada Arduino	34
Gambar 3.25	Program Pemilihan Sektor dan Blok pada Arduino	35
Gambar 3.26	Program Membaca Satu Blok Data dan Satu <i>Byte</i> Data pada <i>Smart Card</i>	35
Gambar 3.27	Program untuk Penulisan <i>Smart Card</i> pada Arduino	36
Gambar 3.28	Menginisialisasi Pin dan Variabel pada Program Pembacaan pada <i>Water Flow Meter</i>	36
Gambar 3.29	Program Penghitungan Pulsa pada Arduino	37
Gambar 3.30	Program Menghitung Kecepatan Aliran Air pada <i>Water Flow Meter</i>	37
Gambar 3.31	Program Menghitung Aliran Air yang Mengalir	38
Gambar 4.1	Rangkaian Pengujian <i>Power Supply</i>	39
Gambar 4.2	Grafik Pengujian Sensor <i>Water Flow Meter</i>	39

DAFTAR TABEL

Tabel 3.1	Tabel Frekuensi RFID <i>Tag</i>	9
Tabel 4.1	Tabel Pengujian Rangkaian <i>Power Supply</i>	40
Tabel 4.2	Tabel Pengujian <i>Water Flow Meter</i> Sensor I	40
Tabel 4.3	Tabel Pengujian <i>Water Flow Meter</i> Sensor II	41
Tabel 4.4	Tabel Pengujian Tegangan dan Waktu pada Sensor <i>Water Flow Meter</i>	42
Tabel 4.5	Tabel Pengujian Data yang Terbaca pada <i>Smart Card</i> ...	44
Tabel 4.6	Tabel Pengujian Pengambilan Data pada <i>Smart Card</i>	44
Tabel 4.7	Tabel Pengukuran Jarak Pembacaan <i>Smart Card</i> RFID Mifare RC522	45
Tabel 4.8	Tabel Pengujian Alat Secara Keseluruhan I	46
Tabel 4.9	Tabel Pengujian Alat Secara Keseluruhan II	47



BAB I

PENDAHULUAN

1.1 Latar Belakang

Salah satu kebutuhan pokok sehari-hari manusia di dunia ini yang tidak dapat dipisahkan adalah air. Tidak hanya penting bagi manusia, air merupakan bagian yang penting bagi makhluk hidup baik hewan dan tumbuhan. Tanpa air kemungkinan tidak ada kehidupan di dunia ini karena semua makhluk hidup sangat memerlukan air untuk bertahan hidup. Manusia bisa bertahan hidup selama beberapa hari tanpa asupan makanan, tetapi manusia tidak akan bertahan selama beberapa hari tanpa asupan air. Karena sudah mutlak bahwa sebagian besar zat pembentuk tubuh manusia itu terdiri dari 73% adalah air^[1]. Selain untuk bertahan hidup, air juga digunakan manusia untuk mandi dan mencuci. Jadi bukan hal yang baru jika kehidupan yang ada di dunia ini dapat terus berlangsung karena tersedianya air yang cukup.

Penggunaan air bersih sangat penting untuk konsumsi rumah tangga, kebutuhan industri dan tempat umum. Karena pentingnya kebutuhan akan air bersih, maka adalah hal yang wajar jika sektor air bersih mendapatkan prioritas penanganan utama karena menyangkut kehidupan orang banyak. Penanganan akan pemenuhan kebutuhan air bersih dapat dilakukan dengan berbagai cara, disesuaikan dengan sarana dan prasarana yang ada. Di daerah perkotaan, sistem penyediaan air bersih dilakukan dengan sistem perpipaan dan non perpipaan. Sistem perpipaan dikelola oleh Perusahaan Daerah Air Minum (PDAM) dan sistem non perpipaan dikelola oleh masyarakat. Kebutuhan air bersih merupakan kebutuhan yang tidak terbatas dan berkelanjutan. Sedangkan, kebutuhan akan penyediaan dan pelayanan air bersih dari waktu ke waktu semakin meningkat yang terkadang tidak diimbangi oleh kemampuan pelayanan. Peningkatan kebutuhan ini disebabkan oleh peningkatan jumlah tempat tinggal penduduk, dan perkembangan kota. Akibat dari meningkatnya jumlah tempat tinggal penduduk yang diiringi peningkatan ekonomi penduduk, peningkatan jumlah kebutuhan air bersih tidak bisa dihindarkan. Kekurangan air pada waktu tertentu

¹Bersumber dari website <http://training.inviro.co.id/fungsi-dan-peran-air-bagi-kehidupan-manusia/> (di akses pada tanggal 22 Oktober 2015)

terutama waktu puncak pemakaian dapat mengganggu kebutuhan air untuk kebutuhan penduduk, sehingga memerlukan alternatif pengaturan dan pendistribusian air secara efektif yang memenuhi kebutuhan minimal pada waktu tersebut. Selain itu, akibat dari meningkatnya jumlah penduduk dapat membuat petugas pengelola air minum atau yang sering disebut PDAM mengalami kesulitan dalam proses pengecekan meteran dari rumah ke rumah yang terkadang pengecekan tidak dilakukan sama sekali. Hal itu mengakibatkan pembayaran tagihan menjadi bertambah pada bulan berikutnya. Hal tersebut dapat menyebabkan kerugian pada masyarakat.

Oleh karena itu, diperlukan inovasi yang dapat mengatasi masalah pengecekan dan pembayaran tagihan PDAM. Dengan cara membuat sebuah *prototype* yang dapat menampilkan data meteran air dengan menggunakan sensor *water flow meter* serta setiap masyarakat juga dapat melakukan pembayaran menggunakan *smart card*. Hal ini akan mengurangi permasalahan yang dialami oleh petugas PDAM dalam melakukan pengecekan setiap bulan. Masyarakat juga dapat melakukan pembayaran dengan mudah dan tidak lagi memikirkan peningkatan biaya tagihan air ketika petugas PDAM tidak melakukan pengecekan meteran air.

1.2 Permasalahan

Pada Tugas Akhir ini yang menjadi permasalahan utama adalah cara menghitung debit air dan cara menggunakan *smart card* untuk pembayaran Perusahaan Daerah Air Minum (PDAM) secara tepat.

1.3 Batasan Masalah

Batasan masalah pada Sistem Penghitungan Debit Air dan Pembayaran menggunakan *Smart Card* pada PDAM adalah alat ini harus ada supply listrik agar dapat digunakan. Jadi apabila listrik padam, air tidak akan bisa mengalir.

1.4 Tujuan

Penelitian ini bertujuan untuk mengetahui cara menghitung debit air menggunakan sensor *water flow meter* dan untuk membuat alat pembayaran yang lebih mudah dengan menggunakan *smart card*. Hasil yang diharapkan dari pelaksanaan tugas akhir ini adalah terciptanya suatu alat yang dapat digunakan untuk menghitung jumlah debit air yang

digunakan pelanggan PDAM serta mengganti pembayaran tagihan bulanan PDAM menjadi pembelian menggunakan *smart card*.

1.5 Metodologi Penelitian

Dalam pelaksanaan tugas akhir yang berupa Sistem Penghitungan Debit Air dan Pembayaran Menggunakan *Smart Card* pada PDAM, ada beberapa kegiatan yang dapat diuraikan sebagai berikut:

a. Tahap persiapan

Pada tahap ini akan dilakukan studi literatur mengenai :

1. Mempelajari karakteristik sensor yang akan digunakan.
2. Mempelajari Karakteristik dari *smart card* yang akan digunakan.
3. Mempelajari kerja dari mikrokontroler ATmega328.
4. Membuat sistem minimum ATmega328.

b. Tahap Perancangan

Pada tahap ini akan dilakukan perancangan sesuai data yang telah didapatkan dari studi literatur.

c. Tahap Pembelian Komponen

Pada tahap ini akan dilakukan pembelian komponen sesuai data yang telah dikumpulkan melalui studi literatur.

d. Tahap Pembuatan

Pada tahap ini akan dilakukan pembuatan alat setelah semua komponen telah lengkap disertai dengan data cara pembuatannya yang diperoleh dari studi literatur.

e. Tahap Pengujian

Pada tahap ini akan dilakukan pengujian alat yang telah dibuat. Di tahap inilah akan diketahui alat bekerja sesuai dengan keinginan atau tidak.

f. Tahap Analisa

Pada tahap ini akan dilakukan analisa. Faktor apa saja yang menyebabkan alat tidak bekerja sesuai dengan keinginan atau terjadi *error*.

g. Tahap Akhir

Pada tahap ini akan dilakukan penyempurnaan pada alat dan membenahi alat jika terjadi *error* sesuai dengan data yang telah didapat pada analisa.

1.6 Sistematika Laporan

Pembahasan Tugas Akhir ini akan dibagi menjadi lima Bab dengan sistematika sebagai berikut:

Bab I Pendahuluan

Bab ini meliputi latar belakang, permasalahan, tujuan penelitian, metodologi penelitian, sistematika laporan, dan relevansi.

Bab II Teori Dasar

Berisi tentang penjelasan teori penunjang yang terkait dengan tugas akhir.

Bab III Perancangan Sistem

Berisi perancangan atau rencana pembuatan yang didahului dengan perancangan perangkat elektronik digunakan sebagai bagian yang melakukan kegiatan, perangkat mekanik yang mendukung terlaksananya tugas akhir, dan perancangan perangkat lunak digunakan untuk memprogram perangkat mekanik dan elektronik agar bekerja sesuai dengan keinginan.

Bab IV Simulasi, Implementasi dan Analisis Sistem

Bab ini memuat hasil simulasi dan implementasi serta analisis dari perangkat lunak dan perangkat elektronik.

Bab V Penutup

Bab ini berisi kesimpulan dan saran dari hasil pembahasan yang telah diperoleh.

1.7 Relevansi

Hasil yang diperoleh dari Tugas Akhir ini diharapkan menjadi referensi untuk mempermudah pembayaran PDAM dan mengurangi resiko yang disebabkan oleh pengecekan meteran PDAM yang tidak rutin.

BAB II TEORI DASAR

2.1 Mikrokontroler ATmega328

Mikrokontroler adalah salah satu dari bagian dasar dari suatu sistem komputer. Seperti umumnya komputer, mikrokontroler adalah alat yang mengerjakan instruksi-instruksi yang diberikan kepadanya. Artinya bagian terpenting dan utama dari suatu sistem terkomputerisasi adalah program itu sendiri. Program ini menginstruksikan komputer untuk melakukan jalinan yang panjang dari aksi-aksi sederhana untuk melakukan tugas yang lebih kompleks. Salah satu mikrokontroler yang dipergunakan pada penelitian ini yaitu ATmega328^[2]. Gambar ATmega 328 dapat dilihat pada gambar 2.1 di bawah ini.



Gambar 2.1 ATmega328

Mikrokontroler ATmega328 menggunakan arsitektur *Harvard* yang memisahkan memori program dari memori data, baik bus alamat maupun bus data. Sehingga pengaksesan program dan data dapat dilakukan secara bersamaan. ATmega328 memiliki fitur cukup lengkap, mulai dari kapasitas memori program dan memori data yang cukup besar, interupsi, *timer* atau *counter*, PWM, dan lainnya. Mikrokontroler ini memiliki beberapa fitur antara lain:

1. Memiliki EEPROM (*Electrically Erasable Programmable Read Only Memory*) sebesar 1KB sebagai tempat penyimpanan data semi permanen karena EEPROM tetap dapat menyimpan data meskipun catu daya dimatikan,

²Bersumber dari website <http://kl601.ilearning.me/2015/10/17/essay-10-arsitektur-mikrocontroller-2/> (di akses pada tanggal 1 Desember 2015)

2. Memiliki SRAM (*Static Random Access Memory*) sebesar 2KB,
3. Memiliki pin I/O digital sebanyak 14 pin 6 diantaranya PWM (*Pulse Width Modulation*) output,
4. 32 x 8-bit *register* serba guna,
5. Dengan *clock* 16 MHz kecepatan mencapai 16 MIPS,
6. 32 KB *Flash Memory* dan pada arduino memiliki *bootloader* yang menggunakan 2 KB dari *flash memory* sebagai *bootloader*, dan
7. 130 macam instruksi yang hampir semuanya dieksekusi dalam satu siklus *clock*.

2.2 RFID Mifare RC522

RFID (*Radio Frequency Identification*) adalah sistem identifikasi tanpa kabel yang memungkinkan pengambilan data tanpa harus bersentuhan seperti *barcode* dan *magnetic card* seperti ATM. Sedangkan, *smart card* adalah sebuah RFID yang bekerja pada frekuensi 13,56 MHz yang dapat digunakan untuk baca dan tulis. Maksud baca dan tulis disini adalah *reader* dapat membaca isi dari *smart card* dan setelah itu menuliskan atau mengganti isi dari *smart card* tersebut. Di dalam sebuah *smart card*, kita bisa memasukkan berbagai informasi. Informasi yang dapat dimasukkan adalah data pribadi, data perusahaan, data transaksi, serta catatan kegiatan yang melibatkan penggunaan kartu tersebut. Media penyimpanan pada *contactless smart card* terdiri atas sejumlah sektor, dan setiap sektor terdiri atas beberapa blok data. Untuk dapat mengakses data, pengguna harus mengetahui alamat sektor yang spesifik.

RFID Mifare RC522 *Reader Module* adalah sebuah modul berbasis IC Philips MFRC522 yang dapat membaca RFID dengan penggunaan yang mudah dan harga yang murah, karena modul ini sudah berisi komponen-komponen yang diperlukan oleh MFRC522 untuk dapat bekerja. Modul ini dapat digunakan langsung oleh MCU dengan menggunakan SPI (*Serial Peripheral Interface*) dengan *supply* tegangan sebesar 3,3V^[3].

³ Bersumber dari website

https://www.addicore.com/v/vspfiles/downloadables/Product%20Downloadables/RFID_RC522/RFIDQuickStartGuide.pdf (diakses pada tanggal 1 Desember 2015)

MFRC522 merupakan produk dari NXP yang menggunakan *fully integrated 13.56MHz non-contact communication card chip* untuk melakukan pembacaan maupun penulisan. Gambar RFID Mifare RC522 dapat dilihat pada Gambar 2.2 dibawah ini.



Gambar 2.2 RFID Mifare RC522

2.3 RFID Tag

RFID *tag* Adalah sebuah alat yang melekat pada obyek yang akan diidentifikasi oleh RFID *reader*. RFID *tag* dapat berupa perangkat pasif atau aktif. *Tag* pasif artinya tanpa *battery* dan *tag* aktif artinya menggunakan *battery*. Tag pasif lebih banyak digunakan karena murah dan mempunyai ukuran lebih kecil. RFID *tag* dapat berupa perangkat *read-only* yang berarti hanya dapat dibaca saja ataupun perangkat *read-write* yang berarti dapat dibaca dan ditulis ulang untuk *update*. RFID *tag* hanya berisi sebuah *tag* yang unik yang berbeda satu dengan yang lainnya. Jadi Informasi mengenai obyek yang terhubung ke *tag* ini hanya terdapat pada sistem atau database yang terhubung pada RFID *reader*^[4]. Gambar RFID tag dapat dilihat pada Gambar 2.3 dibawah ini.



Gambar 2.3 RFID tag

⁴ Bersumber dari website <http://elektronika-dasar.web.id/pengertian-dan-komponen-radio-frequency-identification-rfid/> (diakses pada tanggal 1 Desember 2015)

[illegible]

byte pertama dan 6 *byte* terakhir (kotak hijau) dari tiap blok ini digunakan untuk autentifikasi. Dan pada kotak kuning (4 *byte* pertama block 0) adalah ID (identitas) *tag*. Tiap *tag* mempunyai ID yg berbeda. Oleh karena itu, ketika kita menulis ke blok 0 tidak akan berhasil.

Mifare adalah sebuah *smart card* RFID yang bekerja pada frekuensi 13,56MHz⁵ dapat dilihat pada gambar 2.5 yang ada dibawah ini.

Tabel 2.1 Frekuensi RFID *Tag*

Gelombang	Frekuensi	Rentang Laju Baca	Contoh Penggunaan
LF	125 KHz	±1.5 kaki, kecepatan baca rendah	<i>Access control, animal tracking, point of sale applications</i>
HHF	13.56 MHz	± 3 kaki, kecepatan baca sedang	<i>Acess control, smart cards, item-level tracking</i>
UHF	860-930 MHz	Hingga 15 kaki, kecepatan baca tinggi	<i>Pallet tracking, supply chain management</i>
<i>Microwave</i>	2.45/5.8 Ghz	± 3 kaki, kecepatan baca tinggi	<i>Supply chain management</i>

Pada Tabel 2.1 dijelaskan bahwa gelombang LF (*Low Frequency*), RFID *tag* banyak digunakan untuk identifikasi pada binatang, *keylock* pada mobil dan juga sistem anti pencuri. Frekuensi RFID yang digunakan pada gelombang LF yaitu 125 kHz (standar aslinya). Pada gelombang HHF (*High-frequency*), RFID *tag* sering digunakan untuk *smart card* pada perpustakaan atau toko buku, sebagai akses kontrol pada gedung, pelacakan bagasi pada pesawat terbang dan *item-level tracking*. Frekuensi RFID yang digunakan pada gelombang HHF yaitu

⁵ Bersumber dari website <http://elektronika-dasar.web.id/alokasi-frekuensi-kerja-rfid-radio-frequency-identification/alokasi-frekuensi-kerja-rfid-radio-frequency-identification/> (diakses pada tanggal 10 Desember 2015)

13.56 MHz. Pada gelombang UHF, RFID tag sering digunakan sebagai *pallet tracking* dan *suplay chain management*. Contohnya seperti pelacakan *container*, pelacakan truk dan trailer pada pelabuhan kapal laut. Frekuensi RFID yang digunakan pada gelombang UHF yaitu 860-930 MHz. Pada gelombang *microwave*, RFID tag digunakan sebagai *Supply chain management*. Contohnya RFID tag seringkali digunakan dalam akses kontrol jarak jauh kendaraan bermotor. Frekuensi RFID yang digunakan pada gelombang *microwave* yaitu 2.45/5.8 Ghz.

2.4 Liquid Cristal Display 20x4 (LCD)

Liquid Crystal Display (LCD) adalah suatu jenis media tampil yang menggunakan kristal cair sebagai penampil utama^[6]. LCD sudah digunakan diberbagai bidang misalnya alal-alat elektronik seperti televisi, kalkulator, atau pun layar komputer. Adapun fitur yang disajikan dalam LCD ini terdiri dari 20 karakter dan 4 baris, mempunyai 192 karakter tersimpan, terdapat karakter generator terprogram, dapat dialamati dengan mode 4-bit dan 8-bit yang dilengkapi dengan *back light*. Gambar LCD 20X4 dapat dilihat pada Gambar 2.5 dibawah ini.

Gambar 2.5 LCD 20x4



LCD adalah lapisan dari campuran organik antara lapisan kaca bening dengan elektroda transparan indium oksida dalam bentuk

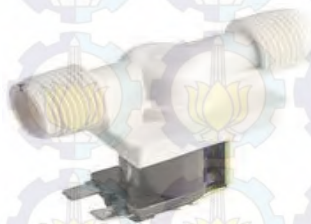
⁶ Bersumber dari website <http://elektronika-dasar.web.id/lcd-liquid-cristal-display/> (diakses pada tanggal 10 Desember 2015)

tampilan *seven-segment* dan lapisan elektroda pada kaca belakang. Ketika elektroda diaktifkan dengan medan listrik (tegangan), molekul organik yang panjang dan silindris menyesuaikan diri dengan elektroda dari segmen. *Register control* yang terdapat dalam suatu LCD diantaranya adalah *register* perintah dan *register* data. *Register* Perintah yaitu *register* yang berisi perintah-perintah dari mikrokontroler ke panel LCD pada saat proses penulisan data atau tempat status dari panel LCD dapat dibaca pada saat pembacaan data. *Register* data yaitu *register* untuk menuliskan atau membaca data dari atau ke DDRAM. Penulisan data pada *register* akan menempatkan data tersebut ke DDRAM sesuai dengan alamat yang telah diatur sebelumnya^[7].

2.5 Solenoid Valve

Solenoid Valve adalah suatu alat kontrol yang berfungsi untuk membuka dan menutup katup atau kran secara otomatis. Kapan *solenoid valve* membuka dan menutup kran ini tergantung dari sensor yang menghubungkan sumber penggeraknya. Sumber penggerak *solenoid valve* bermacam-macam bisa dengan udara yang biasa disebut *pneumatic*, listrik (*electric*) atau gabungan udara dan listrik (*pneumatic electric*).

Berdasarkan tipe bukaan valve ada *Solenoid Valve Normally Closed* (NC) dan *Normally Open* (NO). *Solenoid Valve Normally Closed* adalah pada saat tidak ada tenaga penggerak posisi *valve* akan menutup. Sedangkan, *Solenoid Valve Normally Open* adalah kebalikannya dimana pada saat tidak ada tenaga penggerak posisi *valve* akan terbuka. Gambar *Solenoid Valve* dapat dilihat pada Gambar 2.6 dibawah ini.



Gambar 2.6 *Solenoid Valve*

⁷ Bersumber dari website <https://www.scribd.com/doc/185920131/LCD-20X4> (diakses pada tanggal 10 Desember 2015)

Solenoid Valve bisa dibagi menjadi beberapa bagian yaitu berdasarkan tenaga penggerak ada *Electric Solenoid Valve* dimana tenaga penggeraknya adalah listrik dan *Pneumatic Solenoid Valve* dimana tenaga penggeraknya adalah angin^[8].

2.6 Buzzer

Buzzer adalah sebuah komponen elektronika yang berfungsi untuk mengubah getaran listrik menjadi getaran suara. Pada dasarnya prinsip kerja *buzzer* hampir sama dengan *loud speaker*, jadi *buzzer* juga terdiri dari kumparan yang terpasang pada diafragma dan kemudian kumparan tersebut dialiri arus sehingga menjadi elektromagnet, kumparan tadi akan tertarik ke dalam atau keluar, tergantung dari arah arus dan polaritas magnetnya, karena kumparan dipasang pada diafragma maka setiap gerakan kumparan akan menggerakkan diafragma secara bolak-balik sehingga membuat udara bergetar yang akan menghasilkan suara. *Buzzer* biasa digunakan sebagai indikator bahwa proses telah selesai atau terjadi suatu kesalahan pada sebuah alat (*alarm*)^[9]. *Buzzer* digunakan sebagai penanda apabila token air yang dibeli sudah hampir habis. Gambar *buzzer* dapat dilihat pada gambar 2.7 di bawah ini.



Gambar 2.7 *Buzzer*

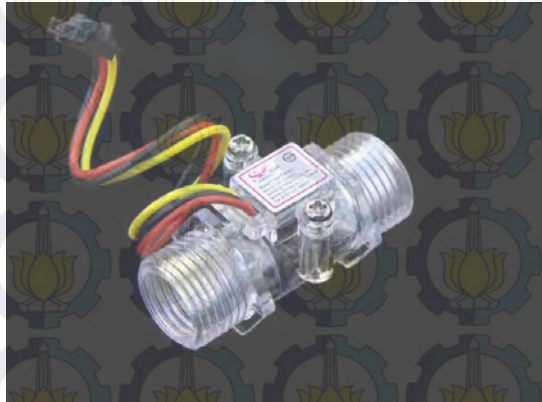
2.7 Water Flow Meter

Water Flow Meter adalah alat untuk mengukur jumlah atau laju aliran dari dalam pipa atau sambungan terbuka. Dengan diketahuinya

⁸ Bersumber dari website <http://dokumen.tips/documents/fungsi-solenoid-valve.html> (diakses pada tanggal 11 Januari 2016)

⁹ Bersumber dari website <http://www.rangkaianelektronika.org/rangkaian-buzzer.htm> (dapat diakses pada tanggal 15 Januari 2016)

parameter dari aliran suatu material oleh alat ukur *water flow meter* yang dikirim berupa data angka dapat juga diteruskan guna menghasilkan aliran listrik atau sinyal yang bisa digunakan sebagai *input* pada *control* atau rangkaian *electric* lainnya. Karakteristik dari fluida yang diukur oleh *water flow meter* sangat luas mulai dari tingkat *corosive* fluida dimana untuk fluida yang tingkat keasamannya tinggi mungkin lebih cocok jika menggunakan *water flow meter* dari bahan PVC atau non logam. Gambar *water flow meter* dapat dilihat pada Gambar 2.8 dibawah ini.



Gambar 2.8 *Water Flow Meter*

Untuk input sensor menggunakan pin 2, dengan menggunakan interrupt 0 pada Arduino uno, penggunaan *interrupt* pada rutin ini dimaksudkan untuk menjalankan program penghitung pulsa yang keluar dari *water flow meter* akibat sensor *hall effect* dan kemudian akan disimpan dalam variabel "*pulseCount*" pada pemrograman arduino^[10].

2.8 Pemrograman Arduino IDE

Arduino adalah pengendali mikro *single-board* yang bersifat *open-source*, diturunkan dari *wiring platform*, dirancang untuk memudahkan penggunaan elektronik dalam berbagai bidang. *Hardware*-nya memiliki prosesor Atmel AVR dan *software*-nya memiliki bahasa pemrograman

¹⁰ Bersumber dari website <https://rudywinoto.com/2012/02/27/461/> (diakses pada tanggal 23 Januari 2016)

sendiri. IDE adalah sebuah *software* untuk menulis program, meng-*compile* menjadi kode biner dan meng-*upload* ke dalam memori mikrokontroler. Mikrokontroler diprogram menggunakan bahasa pemrograman arduino yang memiliki kemiripan *syntax* dengan bahasa pemrograman C. Karena sifatnya yang terbuka maka siapa saja dapat mengunduh skema *hardware* arduino dan membangunnya.

Arduino berevolusi menjadi sebuah *platform* karena ia menjadi pilihan dan acuan bagi banyak praktisi. Tampilan IDE Arduino IDE 1.6.1 bisa dilihat pada gambar 2.10 dibawah ini.



Gambar 2.9 Tampilan Arduino IDE 1.6.1

Saat ini *software* arduino yang digunakan adalah *driver* dan IDE, walaupun masih ada beberapa *software* lain yang sangat berguna selama pengembangan arduino. Tetapi ternyata hasil kompilasi dari arduino *software* dapat dipergunakan dan dijalankan tidak hanya pada papan arduino (*Arduino Boards*) atau turunannya yang kompatibel. Program hasil kompilasi dari arduino dapat dijalankan di sistem mikrokontroler Atmel AVR yang sesuai bahkan tanpa menggunakan *bootloader*. Selain memperluas pilihan varian IC mikrokontroler AVR yang bisa

dipergunakan hal ini juga berarti semakin besar program yang bisa kita muat di *flash memory* IC itu^[11].

2.9 Relay

Relay adalah Saklar yang dioperasikan secara elektrik dan merupakan *switch* komponen *electromechanical* yang terdiri dari 2 bagian utama yakni elektromagnet (*coil*) dan mekanikal (seperangkat kontak saklar). *Relay* menggunakan prinsip elektromagnetik untuk menggerakkan kontak saklar sehingga dengan arus listrik yang kecil (*low power*) dapat menghantarkan listrik yang bertegangan lebih tinggi. Gambar *Relay* dapat dilihat pada Gambar 2.9 dibawah ini.



Gambar 2.10 *Relay*

Kontak poin *relay* terdiri dari 2 jenis yaitu *Normally Close* (NC) yaitu kondisi awal sebelum diaktifkan akan selalu berada di posisi *close* (tertutup) dan *Normally Open* (NO) yaitu kondisi awal sebelum diaktifkan akan selalu berada di posisi *open* (terbuka)¹².

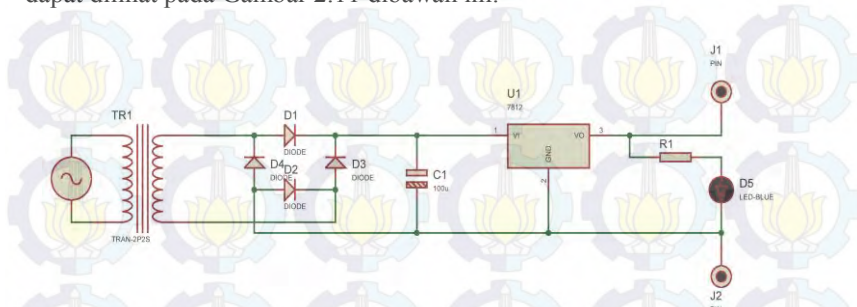
Prinsip kerja dari *relay* yaitu, apabila kumparan (*coil*) diberikan arus listrik, maka akan timbul gaya elektromagnet yang kemudian menarik *armature* untuk berpindah dari posisi sebelumnya (NC) ke posisi baru (NO) sehingga menjadi saklar yang dapat menghantarkan arus listrik di posisi barunya (NO). Posisi dimana *armature* tersebut berada sebelumnya (NC) akan menjadi *open* atau tidak terhubung. Pada saat tidak dialiri arus listrik, *armature* akan kembali lagi ke posisi awal (NC).

¹¹ Bersumber dari website <https://id.wikipedia.org/wiki/Arduino> (diakses pada tanggal 30 Januari 2016)

¹² Bersumber dari website <http://teknikelektronika.com/pengertian-relay-fungsi-relay/> (diakses pada tanggal 10 Januari 2016)

2.10 Power Supply 12V

Power Supply adalah perangkat keras yg berguna untuk menyuplai atau memberi tegangan listrik langsung ke komponen yang membutuhkan tegangan. *Input* daya *power supply* berupa arus bolak-balik (AC) maka *power supply* harus mengubah tegangan AC jadi DC (arus sejalannya). Dalam suatu perangkat elektronika rangkaian *power supply* ada yang menjadi satu kesatuan dengan perangkat elektronik tersebut dan ada juga yang dibuat secara terpisah. Sebuah DC *power supply* atau adaptor pada dasarnya memiliki 4 bagian utama agar dapat menghasilkan arus DC yang stabil. Diagram blok DC *power supply* dapat dilihat pada Gambar 2.11 dibawah ini.



Gambar 2.11 Diagram Blok DC Power Supply

Keempat bagian utama tersebut diantaranya adalah *transformator*, *rectifier*, *filter* dan *voltage regulator*. *Transformator* atau disingkat dengan trafo yang digunakan untuk *power supply* adalah *transformator* jenis *step-down* yang berfungsi untuk menurunkan tegangan listrik sesuai dengan kebutuhan komponen elektronika yang terdapat pada rangkaian *power supply*. *Transformator* bekerja berdasarkan prinsip induksi elektromagnetik yang terdiri dari dua bagian utama yang berbentuk lilitan yaitu lilitan primer dan lilitan sekunder. Lilitan primer merupakan input dari pada *transformator* sedangkan *output* -nya adalah pada lilitan sekunder. Meskipun tegangan telah diturunkan, *output* dari *transformator* masih berbentuk arus bolak-balik yang harus diproses selanjutnya. *Rectifier* atau penyearah gelombang adalah rangkaian elektronika dalam *power supply* yang berfungsi untuk mengubah gelombang AC menjadi gelombang DC setelah tegangannya diturunkan oleh *transformator step-down*. Rangkaian *rectifier* biasanya terdiri dari komponen dioda. Dalam rangkaian *Power supply*, *filter* digunakan untuk

meratakan sinyal arus yang keluar dari *rectifier*. Untuk menghasilkan tegangan dan arus DC (arus searah) yang tetap dan stabil, diperlukan *voltage regulator* yang berfungsi untuk mengatur tegangan sehingga tegangan *output* tidak dipengaruhi oleh suhu, arus beban dan juga tegangan *input* yang berasal *output filter*. *Voltage regulator* pada umumnya terdiri dari diode zener, transistor atau *Integrated Circuit* (IC)^[13].

2.11 Tarif Air pada PDAM

Tarif Air PDAM diklasifikasikan berdasarkan Peraturan Perusahaan, Perusahaan Daerah Air Minum Kota Surabaya No. 04 Tahun 2008 Tanggal 03 Maret 2008. Tabel Tarif Air pada PDAM dapat dilihat pada Gambar 2.12 dibawah ini.

KLASIFIKASI	Kode Tarif	Pemak. Air (M ³)	Tarif Air (Rp/M ³)	Pemak. Min/Bin (M ³)
1	2	3	4	5
KELOMPOK PELANGGAN IV	3A	0 - 10 11 - 20 >20	500 1.200 1.900	10
1. Rumah tangga (RT) 2, yaitu : Kelompok pelanggan rumah tangga yang tidak memenuhi salah satu kriteria RT3, RT4,RT5 dan memenuhi salah satu kriteria sebagai berikut : a. Didepannya terdapat jalan dengan lebar termasuk saluran/got dan berm ≥ 3 meter akan tetapi < 5 meter; b. Daya listrik yang terpasang < 1300 VA; c. Nilai Jual Obyek Pajak (NJOP) \geq Rp.50 juta akan tetapi $< Rp.150$ juta; d. Luas bangunan ≥ 36 m ² akan tetapi < 120 m ² .				

Gambar 2.12 Tabel Tarif Air pada PDAM

¹³ Bersumber dari website <http://teknikelektronika.com/prinsip-kerja-dc-power-supply-adaptor/> (diakses pada tanggal 5 Februari 2016)

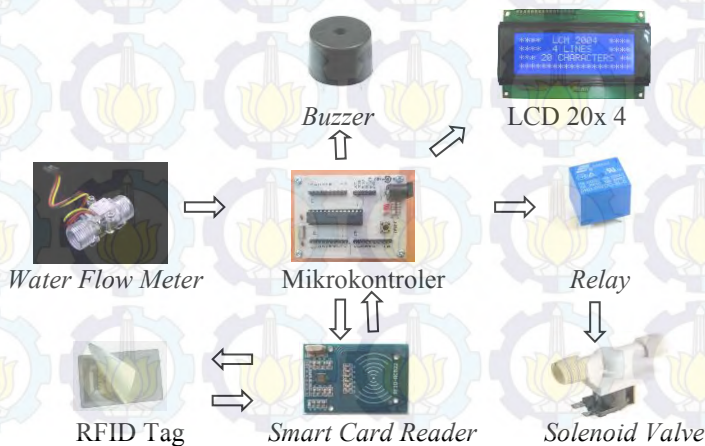
Tarif air yang digunakan pada kelompok pelanggan IV memiliki minimal pemakaian per bulan sebanyak 10 m³. Dan tarif per m³ berbeda-beda berdasarkan pemakaian air sesuai dengan Gambar 2.12^[14].

¹⁴Bersumber dari website http://www.pdam-sby.go.id/page.php?get=tampil_tabel_tarif&bhs=1 (diakses pada tanggal 10 Juni 2016)

BAB III

PERANCANGAN SISTEM KONTROL

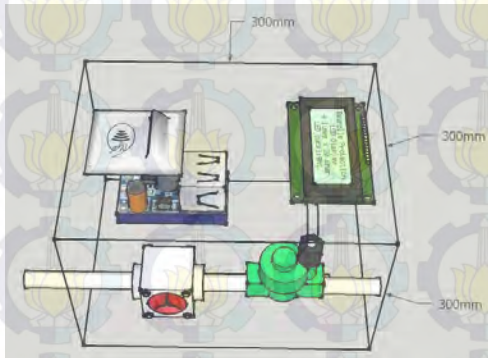
Pada perancangan alat Sistem Penghitungan Debit Air dan Pembayaran Menggunakan *Smart Card* pada PDAM terdapat 3 bagian penting yang meliputi perancangan dan pembuatan mekanik, perancangan dan pembuatan perangkat keras (*hardware*), serta perancangan dan pembuatan perangkat lunak (*software*). Pada perancangan dan pembuatan mekanik dengan cara membuat 2 tempat penampungan air yang salah satunya ditempatkan lebih tinggi agar air bisa mengalir. Untuk menyambungkan keduanya, diberi pipa pada bagian bawah untuk penampungan air yang lebih tinggi dan pada bagian atas untuk penampungan air yang lebih rendah. Pada perancangan perangkat keras (*hardware*) akan dibuat beberapa rangkaian, diantaranya yaitu rangkaian minimum sistem mikrokontroler ATmega328, rangkaian *power supply*, rangkaian *relay* untuk *solenoid valve*, rangkaian *buzzer* sebagai indikator, rangkaian *Smart Card Reader* yang menggunakan MFRC522, dan rangkaian LCD 20x4. Sedangkan untuk perancangan dan pembuatan perangkat lunak (*software*) meliputi pembuatan program mikrokontroler ATmega328 dengan menggunakan program arduino. Diagram Fungsional Pembuatan Perangkat Elektronik terdapat pada Gambar 3.1 dibawah ini.



Gambar 3.1 Diagram Fungsional Pembuatan Perangkat Elektronik

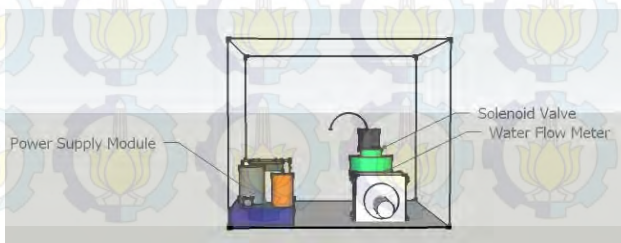
3.1 Perancangan dan Pembuatan Mekanik

Perancangan mekanik Sistem Penghitungan Debit Air dan Pembayaran Menggunakan *Smart Card* pada PDAM yaitu membuat dua buah tempat air berbentuk seperti kubus yang digunakan sebagai penampungan air. Lalu, air akan mengalir dari satu penampungan ke penampungan yang lainnya melalui sebuah pipa dengan cara menempatkan salah satu penampungan air pada tempat yang lebih tinggi. *Hardware* yang digunakan untuk menghitung debit air akan ditempatkan diantara dua penampungan air tersebut. Pada gambar berikut ini, adalah gambar perancangan mekanik yang digunakan sebagai tempat dari *hardware* yang digunakan. Gambar Perancangan Mekanik Tampak Atas dapat dilihat pada Gambar 3.2 dibawah ini.



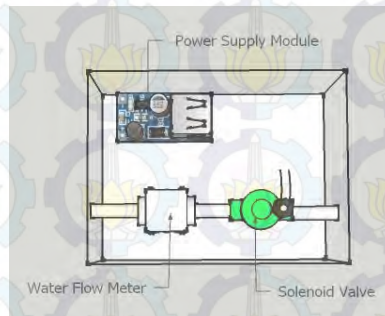
Gambar 3.2 Perancangan Mekanik Tampak Atas

Pada Gambar 3.2 terdapat LCD dan *smart card reader* yang berada pada atas alat. Gambar Perancangan Mekanik Tampak Samping dapat dilihat pada Gambar 3.3 dibawah ini.



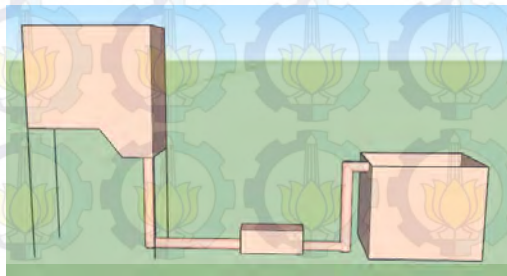
Gambar 3.3 Perancangan Mekanik Tampak Samping

Pada gambar 3.3, terdapat satu buah sensor *water flow meter*, *solenoid valve*, DC *power supply* 12 V dan pipa penghubung yang dapat dilihat dari samping. Dimana pipa tersebut akan dihubungkan ke bak air yang berada diatas dan bak air yang berada dibawah. Gambar Perancangan Mekanik Tampak Dalam dapat dilihat pada Gambar 3.4 dibawah ini.



Gambar 3.4 Perancangan Mekanik Tampak Dalam

Pada gambar 3.4, terdapat *hardware* yang digunakan yaitu DC *power supply* 12V pada gambar yang berwarna abu-abu, sensor *water flow meter* yang terdapat pada bagian kiri dari pipa, *solenoid valve* yang berada di kanan pipa, pipa untuk menyambung antara *solenoid valve* dan *water flow meter*. *Water flow meter* sendiri ditempatkan sebelum *solenoid valve* agar pada *water flow meter* terdapat air dan tidak menyebabkan *water flow meter* berputar jika *solenoid valve* dalam keadaan tertutup (*close*). Gambar Perancangan Mekanik Keseluruhan dapat dilihat pada Gambar 3.5 dibawah ini.



Gambar 3.5 Perancangan Mekanik Keseluruhan

Pada gambar 3.5, terdapat dua bak air yang besar. Dua bak air besar tersebut berfungsi sebagai tempat penyimpanan air. Perbedaan penempatan antara dua bak air tersebut berfungsi agar air dapat mengalir seperti air pada PDAM. Untuk menyambungkan keduanya, terdapat pipa yang berada pada masing-masing sisi. Pada bagian tengah, terdapat sebuah *box* kecil. *Box* kecil inilah yang berisi *hardware* yang digunakan sebagai alat penghitungan debit air dan pembayarannya menggunakan *smart card*.

3.2 Perancangan dan Pembuatan Perangkat Elektronik

Blok diagram dari keseluruhan sistem kendali atau perangkat elektronik pada Sistem Penghitungan Debit Air dan Pembayaran Menggunakan *Smart Card* pada PDAM yang didesain pada tugas akhir ini. Perancangan *hardware* dilakukan dengan merancang rangkaian-rangkaian elektronika.

Perancangan *hardware* ini meliputi :

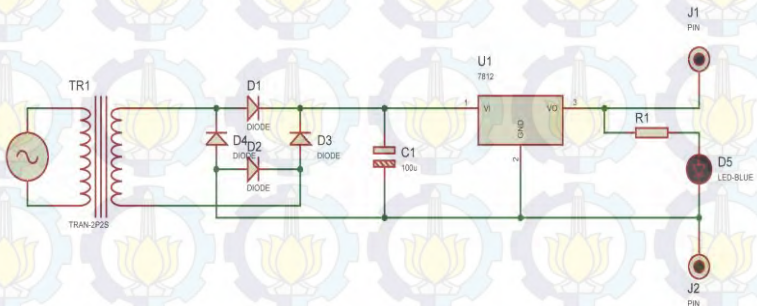
1. Rangkaian *Power Supply*
2. Rangkaian Minimum Sistem ATmega328
3. Rangkaian *Relay* dan *Solenoid Valve*
4. Rangkaian *Smart Card*
5. Rangkaian LCD 20x4
6. Rangkaian *Buzzer*

Alat ini dibuat untuk mengatasi masalah pengecekan dan pembayaran tagihan PDAM. Alat ini dapat menampilkan data debit air yang mengalir menggunakan sensor *water flow meter* dan dapat melakukan pembayaran dengan menggunakan *smart card*. Semua komponen yang digunakan diolah oleh *cip* mikrokontroler ATmega328. Untuk menampilkan hasil dari penghitungan sensor *water flow meter*, digunakan rangkaian LCD 20x4. Saat pulsa dari alat ini akan habis, maka *buzzer* akan berbunyi sebagai penanda.

3.2.1 Rangkaian *Power Supply*

Rangkaian *power supply* digunakan untuk menyediakan tegangan DC yang berfungsi untuk seluruh kebutuhan sumber energi yang dibutuhkan pada alat yang akan digunakan. Untuk alat yang dibuat, digunakan sebuah *power supply* dengan buah keluaran. Setiap keluaran dari *power supply* memiliki tegangan keluaran sebesar, 12 Volt. Skematik rangkaian dan *board power supply* dengan keluaran 12 Volt. *Power supply* bertegangan 12 Volt digunakan untuk meng-supply

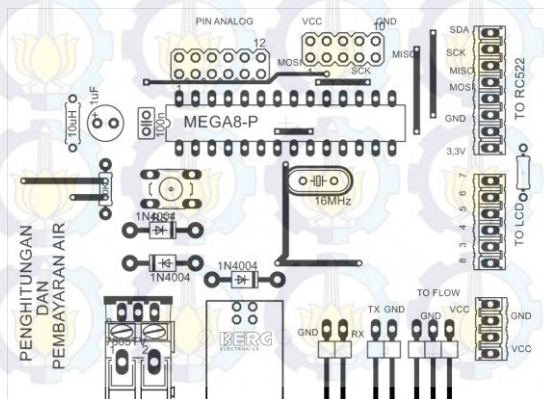
ATmega328 dan *solenoid valve*. Sedangkan, untuk meng-supply RC522 dan *water flow meter* menggunakan *output* tegangan pada minimum sistem. Gambar Rangkaian *Power Supply* dapat dilihat pada Gambar 3.6 dibawah ini.



Gambar 3.6 Rangkaian *Power Supply*

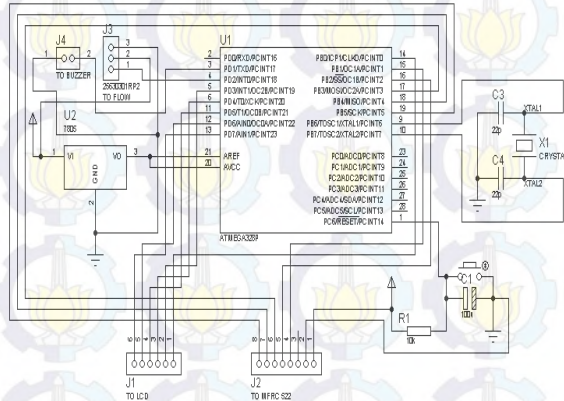
3.2.2 Rangkaian *Minimum System* ATmega328

Perancangan rangkaian ini digunakan sebagai otak dari alat yang dibuat. Rangkaian *minimum system* akan disambungkan ke semua rangkaian atau komponen yang digunakan. Yaitu *power supply*, *water flow meter*, *buzzer*, *relay*, *solenoid valve*, RC522 (*smart card reader*), dan LCD. Gambar *Board Minimum System* ATmega328 dapat dilihat pada Gambar 3.7 dibawah ini.



Gambar 3.7 *Board Minimum System* ATmega328

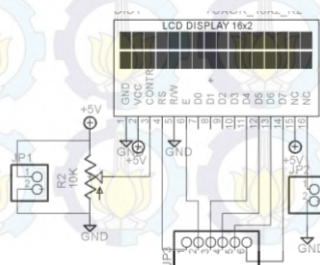
Rangkaian *minimum system* ATmega 328 yang akan digunakan pada tugas akhir ini dapat dilihat pada Gambar 3.8 dibawah ini.



Gambar 3.8 Rangkaian *Minumum System* ATmega328

3.2.3 Rangkaian LCD

Perancangan rangkaian LCD dimaksudkan untuk tampilan karakter yang diinginkan pada LCD 20x4. Rangkaian LCD 20x4 terhubung dengan minimum sistem mikrokontroler ATmega328 pada pin RS, pin E, pin D4, pin D5, pin D6, dan pin D7. Selain itu, pin VCC dan pin GND juga disambungkan ke mikrokontroler ATmega328. Karakter yang ditampilkan pada LCD meliputi pembacaan data dari *water flow meter* dan *smart card* yang isinya tentang berapa liter air yang digunakan dan berapa liter sisa air yang dapat digunakan. Gambar Rangkaian LCD dapat dilihat pada Gambar 3.9 dibawah ini.

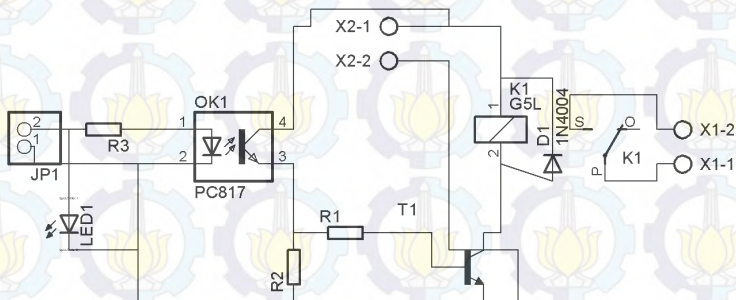


Gambar 3.9 Rangkaian LCD

3.2.4 Rangkaian Driver Relay

Rangkaian *driver relay* digunakan untuk mengaktifkan *solenoid valve* yang digunakan untuk membuka atau menutup aliran air. *Relay* yang digunakan memiliki 5 pin. 2 pin *coil* yaitu *coil* positif dan *coil* negatif yang menghubungkan induktor didalam *relay*. Dan 3 buah pin yang lain yaitu pin *normally open*, pin *normally close*, dan pin *common* terminal.

Dalam rangkaian *relay* terdapat sebuah dioda yang terpasang paralel dengan *relay*. Fungsi dari dioda tersebut adalah untuk mengalirkan arus yang lewat ketika *switch* dimatikan. Transistor pada rangkaian *relay* digunakan sebagai *switch* yang dihubungkan dengan mikrokontroler sebagai pemberi perintah. Gambar Skema Rangkaian *Relay* terdapat pada Gambar 3.10 dibawah ini.



Gambar 3.10 Skema Rangkaian *Relay*

3.3 Perancangan dan Pembuatan Perangkat Lunak (Software)

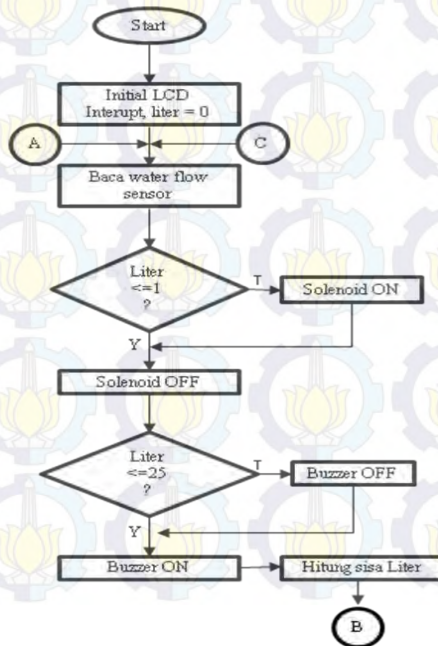
Perancangan perangkat lunak dalam alat ini adalah membuat program untuk menghitung jumlah air yang mengalir dalam bentuk liter dan dapat menghentikan aliran air menggunakan *solenoid valve* jika penghitungan telah mencapai hasilnya. Air akan kembali mengalir jika pada alat ini telah diisi pulsa menggunakan *smart card*. Perangkat lunak yang digunakan yaitu arduino. Perangkat lunak arduino diprogram agar mikrokontroler ATmega328 dapat menghitung jumlah debit air pada sensor *water flow meter* dan menggerakkan *solenoid valve* menggunakan *driver relay*. Sebelum menggunakan minimum sistem ATmega328 dengan menggunakan perangkat lunak arduino, terlebih dahulu harus mengisikan *bootloader* pada ATmega328 yang akan

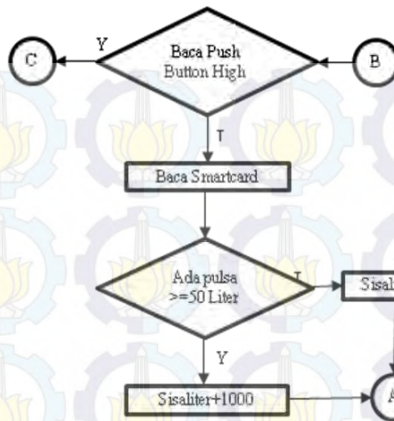
digunakan. Cara yang paling mudah mengisikan *bootloader* pada ATmega328 adalah menggunakan arduino uno. Langkah mengisikan *bootloader* ATmega328 yaitu:

1. Menghubungkan arduino uno dengan komputer dengan kabel USB dan minimum sistem ATmega328 dengan menggunakan pin mosi, miso, sck, rst, gnd dan vcc.
2. Membuka perangkat lunak IDE Arduino, lalu memilih *File-Examples-ArduinoISP*.
3. Memilih jenis *board* yang digunakan pada *tools* yaitu arduino uno dan *programmer* yaitu arduino ad ISP
4. Meng-*upload* program yang telah di *setting*
5. Setelah selesai *upload*, kemudian memilih *tools-burn bootloader*.

3.3.1 Pembuatan FlowChart

Diagram alur (*FloChart*) sistem dapat dilihat pada Gambar 3.11 dibawah ini.





Gambar 3.11 *Flowchart System*





Diagram alur (*Flowchart*) sistem dimulai dengan pembacaan data saldo yang ada di mikrokontroler. Apabila data liter air pada mikrokontroler masih ada, maka alat dapat langsung digunakan. Sedangkan apabila pada mikrokontroler data liter air telah habis, maka terlebih dahulu harus mengisi saldo menggunakan *smart card*. Caranya dengan menekan tombol *push button* yang telah disediakan, lalu memasukkan *smart card* ke *smart card reader* akan otomatis membaca dan mengurangi isi dari *smart card* sesuai dengan program yang digunakan. Pengurangan pada isi *smart card* adalah 1000 liter. Jika pada *smart card* memiliki isi kurang dari 1000 liter, maka liter sisa pada mikrokontroler akan bertambah sebesar 0 liter atau tidak bertambah. Pengisian saldo pada mikrokontroler menggunakan *smart card* dapat dilakukan kapan saja. Meskipun saldo pada mikrokontroler masih banyak, saldo tetap bisa dimasukkan. Penghitungan sisa saldo dan penghitungan besar liter air yang mengalir pada sensor *water flow meter* akan ditampilkan pada LCD 20x4 yang digunakan pada alat. Saat sisa saldo liter air memiliki nilai kurang dari sama dengan 0 Liter, maka air tidak bisa mengalir karena terdapat *solenoid valve* yang tertutup. *Solenoid valve* yang digunakan sendiri adalah *solenoid valve normally close*. Hal ini difungsikan ketika terjadi pemadaman listrik, maka air tidak akan mengalir. Ketika sisa saldo pada mikrokontroler kurang dari atau sama dengan 25 Liter, maka *buzzer* akan menyala. Ini menandakan bahwa sisa saldo liter air sudah mendekati habis.

Pada pembuatan perangkat lunak (*software*) ini menggunakan *software* arduino seperti pada Gambar 3.12 dibawah ini.





Gambar 3.12 Tampilan *Software* Arduino

Dari Gambar 3.12 menunjukkan *software* arduino versi 1.6.1 yang akan digunakan untuk membuat program pada tugas akhir ini. Sebelum menjelaskan langkah-langkah pembuatan program pada tugas akhir ini, maka terlebih dahulu di jelaskan beberapa *icon* yang sering di gunakan, yaitu :

1.  Merupakan *icon* *create new project*. *Icon* ini berfungsi untuk memulai sebuah projek program.
2.  Adalah *icon* menu *verify* yang bergambar *checklist*, ini berfungsi untuk mengecek program yang ditulis apakah ada yang salah atau *error*.
3.  Adalah *icon* menu *upload* yang bergambar panah ke arah kanan, ini berfungsi untuk memuat atau meng-*transfer* program yang dibuat di *software* arduino ke *hardware* arduino.
4.  Adalah *icon* menu *open* yang bergambar panah ke arah atas ini berfungsi untuk membuka program yang disimpan

atau membuka program yang sudah dibuat dari pabrik *software* arduino.

5.  Adalah *icon* *save* yang bergambar panah ke arah bawah, ini berfungsi untuk menyimpan program yang telah dibuat atau dimodifikasi
6.  Adalah *icon* menu *serial monitor* yang bergambar kaca pembesar (*loop*), ini berfungsi mengirim atau menampilkan *serial* komunikasi data saat dikirim dari *hardware* arduino.

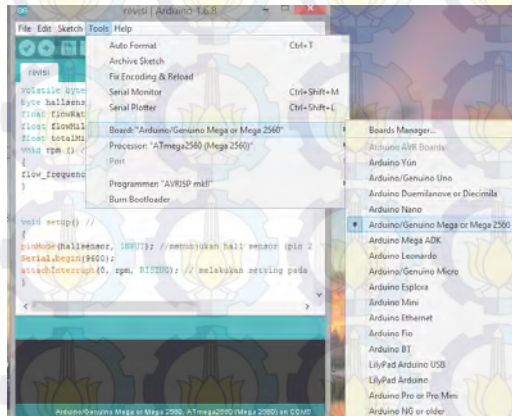
Setelah mengetahui beberapa *icon* yang digunakan dan telah membuat program dalam *software*, maka langkah selanjutnya yang harus dilakukan adalah cara meng-*upload* program pada Arduino. Untuk meng-*upload* program pada arduino terdapat langkah-langkah yang harus dilakukan.

Berikut langkah-langkah meng-*upload* program pada arduino. Langkah pertama yang dilakukan setelah membuat program adalah melakukan *verify* program yang digunakan dengan cara menekan tombol *verify*. Jika tidak ada *error* maka akan muncul kata *done compiling* seperti gambar 3.13 dibawah ini.



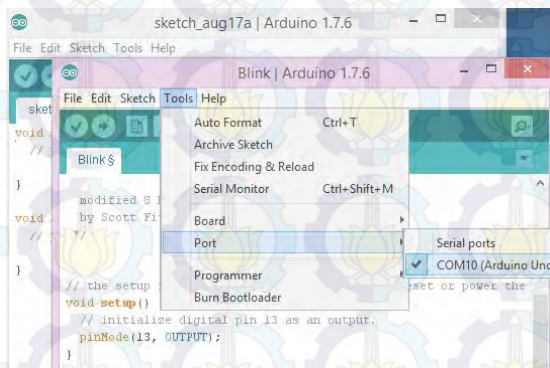
Gambar 3.13 Tampilan Arduino Ketika Tidak Terjadi *Error* pada Saat *Verify* Program

Langkah kedua adalah menyambungkan *hardware* arduino dengan laptop atau pc yang digunakan. Langkah ketiga adalah memilih *board* arduino yang digunakan dengan cara menekan *tools* lalu *board* dan *board* yang digunakan seperti pada gambar 3.14 dibawah ini.



Gambar 3.14 Memilih Board Arduino

Langkah keempat adalah memilih *port* yang digunakan pada laptop atau PC seperti pada gambar 3.15 dibawah ini.



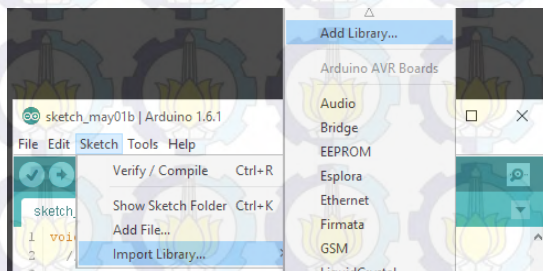
Gambar 3.15 Memilih Port pada Arduino

Langkah terakhir adalah menekan *icon upload* atau menekan *file* → *upload*.

3.3.2 Perancangan Program pada Arduino

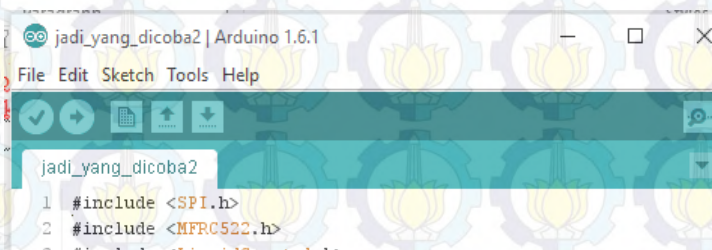
Pada tahapan ini, menggunakan *library* MFRC522 pada arduino yang terlebih dahulu harus di *download* pada *website* arduino.

Berikut ini adalah langkah-langkah perancangan program pada *smart card* diarduino. Langkah pertama adalah mencari *library* MFRC522 pada arduino di *website* arduino dan memasukkannya pada *software* arduino dengan cara memilih *sketch* lalu *import file* dan *add library* pada arduino seperti gambar 3.16 dibawah ini. Lalu memilih *file* dalam bentuk ZIP yang telah di *download*.



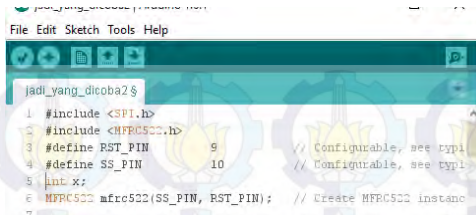
Gambar 3.16 Menambah *Library* MFRC522 pada Arduino

Setelah *library* di tambahkan pada *software*, langkah selanjutnya adalah membuat program dengan terlebih dahulu memasukkan *library* yang telah ditambahkan seperti pada gambar 3.17 dibawah ini.



Gambar 3.17 Menambahkan *Library* pada Arduino

Langkah selanjutnya adalah membuat *listing* program yang digunakan untuk *smart card*. Sebelum pembuatan program ini, harus menginisialisasi pin yang digunakan seperti pada Gambar 3.18 dibawah ini.



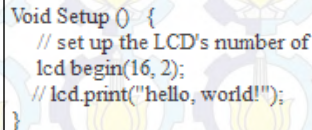
```

1 #include <SPI.h>
2 #include <M5Stack.h>
3 #define RST_PIN 9 // Configurable, see typical pin layout
4 #define SS_PIN 10 // Configurable, see typical pin layout
5 int x;
6 M5Stack m5(M5STACK_SS_PIN, M5STACK_RST_PIN); // Create M5Stack instance

```

Gambar 3.18 Menginisialisasi Pin pada Arduino

Selanjutnya adalah masuk pada program *void setup*. Program *void setup* hanya sekali eksekusi yaitu ketika awal program berjalan. Kebanyakan *void setup* berisikan inisialisasi fungsi-fungsi yang digunakan dalam program. *Void setup* juga dapat digunakan sebagai penggunaan kaki dari arduino. *Void setup* ini merupakan fungsi wajib yang harus disertakan dalam memprogram arduino, jika tidak disertakan maka akan menghasilkan *error*. Isi dari *void setup* seperti pada gambar 3.19 dibawah ini.



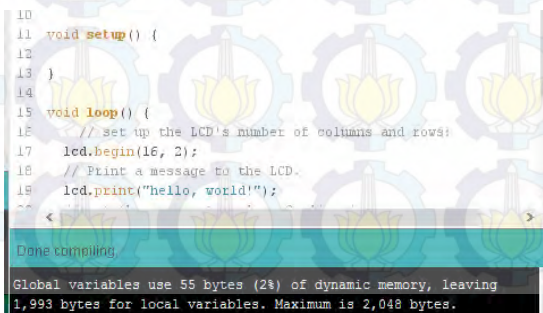
```

Void Setup () {
    // set up the LCD's number of
    lcd.begin(16, 2);
    // lcd.print("hello, world!");
}

```

Gambar 3.19 Program *Void Setup* pada Arduino

Selain itu, Isi *void setup* bisa juga kosong seperti pada Gambar 3.20 dibawah ini.



```

10
11 void setup() {
12
13 }
14
15 void loop() {
16     // set up the LCD's number of columns and rows:
17     lcd.begin(16, 2);
18     // Print a message to the LCD.
19     lcd.print("hello, world!");
20
21 }

```

Done compiling.

Global variables use 55 bytes (2%) of dynamic memory, leaving 1,993 bytes for local variables. Maximum is 2,048 bytes.

Gambar 3.20 *Void Setup* yang kosong pada Arduino

Selanjutnya memasuki perintah *void loop*. *Void loop* berguna untuk melaksanakan atau mengeksekusi perintah program yang telah dibuat. Fungsi ini akan secara aktif mengontrol *board* arduino baik membaca *input* atau merubah *output*. Setelah itu, tinggal melakukan *verify* program dan program telah siap untuk di-*upload* ke *board* arduino.

3.3.3 Perancangan Program Smartcard

Untuk merancang program *smart card* pada arduino, terlebih dahulu harus memasukkan *library* yang digunakan yaitu *library* MFRC522 dan *library* SPI untuk komunikasi *serial*-nya. Selanjutnya adalah melakukan inisialisasi pin-pin yang digunakan pada *board* arduino. Proses pengiriman dan penerimaan data antara *smart card* pada arduino menggunakan pin-pin digital. Pin digital yang digunakan adalah pin 9,10,11,12, dan 13. Pin yang didefinisikan hanya 9 dan 10, karena yang 11,12, dan 13 adalah pin mosi, miso dan sck, jadi tidak memerlukan pendefinisian. Inisialisasi pin yang digunakan pada *board* arduino dapat dilihat pada gambar 3.21 dibawah ini.

```
#include <SPI.h>
#include <MFRC522.h>
#define RST_PIN      9
#define SS_PIN       10
MFRC522 mfrc522(SS_PIN, RST_PIN);
```

Gambar 3.21 Inisialisasi Pin pada Board Arduino

Selanjutnya adalah melakukan pemrograman fungsi *setup* pada Arduino. Fungsi *setup* pada program *smart card* berisi tentang inisialisasi *smart card* dan komunikasi *serial*. Pemrograman fungsi *setup* dapat dilihat pada Gambar 3.22 dibawah ini.

```
Void Setup () {
    Serial.begin (9600);
    while (!Serial);
    SPI.begin ();
    mfrc522.PCD_Init();

    Serial.print("Scan a MIFARE Classic");
    delay(1000);
}
```

Gambar 3.22 Pemrograman Fungsi *setup* pada Arduino

Prosedur selanjutnya adalah perintah yang digunakan untuk memilih salah satu *byte* data dari sebuah blok pada *smart card*. Perintah itu dapat dilihat pada Gambar 3.23 dibawah ini.

```
void dump_byte_array(byte *buffer, byte bufferSize) {  
    for (byte i = 0; i < bufferSize; i++) {  
        Serial.print (buffer[i] < 0x10 ? " 0" : " ");  
        Serial.print (buffer[i], DEC) ;  
    }  
}
```

Gambar 3.23 Memilih Salah Satu *Byte* Data dari Sebuah Blok pada *Smartcard*

Data yang dihasilkan dapat berupa *hexadecimal* atau *decimal*. Hal ini tergantung dengan perintah yang digunakan. Setelah semua yang dibutuhkan telah dibuat, prosedur selanjutnya adalah membuat program inti yaitu program yang digunakan untuk membaca dari *smart card*, melakukan pengurangan data yang telah dibaca, dan dilanjutkan dengan pengiriman data pada *smart card*. Untuk pembacaan dari *smart card*, terdapat beberapa perintah yang digunakan. Langkah pertama dapat dilihat pada Gambar 3.24 dibawah ini.

```
void loop() {  
    // Look for new cards  
    if (! mfrc522.PICC_IsNewCardPresent())  
        return;  
  
    // Select one of the cards  
    if (! mfrc522.PICC_ReadCardSerial())  
        return;  
  
    // Show some details of the PICC (that is: the tag/card)  
    led.setCursor(0,0);  
    lcd.print("type Kartu")  
    MFRC522::PICC_Type piccType = mfrc522.PICC_GetType (mfrc522.uid.sak) ;  
    lcd.setCursor(10,0);  
    lcd.print(mfrc522.PICC_GetTypeName(piccType));  
  
    // Check for compatibility  
    if ( piccType != MFRC522::PICC_TYPE_MIFARE_MINI  
        " piccType != MFRC522::PICC_TYPE_MIFARE_1K  
        " piccType != MFRC522::PICC_TYPE_MIFARE_4K) (  
        SerialPrintln(F("This sample only works with MIFARE Classic cards.)) ;  
        return;  
    }  
}
```

Gambar 3.24 Program untuk Membaca *Smartcard* pada Arduino

Pada Gambar 3.24 menunjukkan untuk pembacaan *type* kartu yang digunakan. Ada beberapa *type* kartu yang dapat digunakan tetapi tidak semua dari *type* kartu yang bisa dibaca oleh *reader*. Pada program, terdapat beberapa *type* kartu yang mendukung dengan *reader* yang digunakan. *Type* kartu yang mendukung adalah *mifare mini*, *mifare 1k*, dan *mifare 4k*. Langkah kedua yaitu program pemilihan sektor dan blok dapat dilihat pada Gambar 3.25 dibawah ini.

```
byte sector      = 1;
byte blockAddr   = 4;
byte trailerBlock = 7;
MFRC522::StatusCode status;
byte buffer[18];
byte size = sizeof(buffer);
```

Gambar 3.25 Program Pemilihan Sektor dan Blok pada Arduino

Pada Gambar 3.25 dapat dilihat bahwa menentukan tempat pembacaan sektor dan blok yang diinginkan. Jadi pada *smart card* ini, terdapat 16 sektor, dan setiap satu sektornya terdapat 4 blok. Tetapi yang dapat digunakan hanya 3 blok untuk menyimpan data. Sedangkan, semua sektor dan semua blok tetap dapat dibaca oleh *reader*. Jika data pada blok yang tidak dapat menyimpan data digantikan, maka 4 blok dalam 1 sektor yang datanya digantikan tersebut tidak dapat digunakan lagi. Potongan program dapat dilihat pada Gambar 3.26 dibawah ini.

```
lcd.setCursor(0,0);
lcd.print("Data pada Block"); Serial.print(blockAddr);
lcd.setCursor(0,1);
lcd.print("byte 0");
status = (MFRC522::StatusCode) mfr522.MIFARE_Read(blockAddr, buffer, &size);
x=buffer[0];
lcd.setCursor(8,1);
lcd.print(x);
if (status != MFRC522::STATUS_OK) {
    lcd.setCursor(8,1);
    lcd.print("MIFARE_Read() failed: ");
    lcd.print((mfr522.GetStatusCodeName(status)));
}

lcd.setCursor(0,2);
lcd.print("Terbaca");
lcd.setCursor(9,2);
lcd.print(buffer[0]);
lcd.setCursor(12,2);
lcd.print("00 LITER");
dataBlock[0]=x-10;
lcd.setCursor(0,3);
lcd.print("dikurangi 1000L");
```

Gambar 3.26 Program Membaca Satu Blok Data dan Satu *Byte* Data pada *Smartcard*

Pada Gambar 3.26 menunjukkan untuk pembacaan 1 blok data dan 1 byte data pada *smart card*. Yang digunakan adalah 1 byte data. 1 byte data dibaca menggunakan perintah *buffer* yang telah dideklarasikan pada *dumb byte array*. Setelah program terbaca, maka data pada byte tersebut akan dikurangi oleh program. Pengurangan ini bertujuan untuk melakukan pembayaran menggunakan *smart card*.

Ketika *byte* telah dikurangi oleh program, maka hasilnya akan dituliskan lagi ke *byte data* yang sebelumnya telah dibaca. Maka data akan otomatis berubah dan isinya akan berkurang sesuai program yang digunakan. *Byte data* memiliki nilai 255 atau ff dalam *hexadecimal*. Untuk menulis *byte data*, harus menggunakan 1 blok. Jadi ketika yang dibutuhkan hanya 1 *byte data*, maka harus menuliskan 1 blok. Program untuk penulisan *smar tcard* dapat dilihat pada Gambar 3.27 dibawah ini.

```
Serial.print(F("Writing data into block")); Serial.print (blockAddr);
Serial.println(F("....."));
dump_byte_array(dataBlock, 16); Serial.println();
status = (MFRC522::StatusCode) mfrc522.MIFARE_Write(blockAddr, dataBlock, 16);
if (status != MFRC522::STATUS_OK) {
    Serial.print (F("MIFARE_Write() failed: "));
    Serial.println(mfrc522.GetStatusCodeName(status));
}
```

Gambar 3.27 Program untuk Penulisan *Smartcard* pada Arduino

3.3.4 Perancangan Program *Flow Meter*

Langkah pertama yang dilakukan untuk merancang program arduino untuk *water flow meter* sensor adalah menginisialisasi pin dan *variable* yang digunakan pada program pembacaan *water flow meter*. Programnya dapat dilihat pada Gambar 3.28 dibawah ini.

```
1 volatile byte flow_frequency; //frekuensi pulsa
2 byte hallsensor = 2;
3 float flowRate;
4 float flowMillilitres;
5 float totalMillilitres;
6
7 void rpm () //untuk memanggil interrupt
```

Gambar 3.28 Menginisialisasi Pin dan Variabel pada Program Pembacaan *Water Flow Meter*

Untuk *input* sensor menggunakan pin 2, dengan menggunakan interrupt 0 pada arduino uno, penggunaan *interrupt* pada rutin ini dimaksudkan untuk menjalankan program penghitung pulsa yang keluar dari *water flow sensor* akibat sensor *hall effect* dan kemudian akan disimpan dalam variabel "*flow_frequency*". Fungsinya adalah untuk menyimpan penghitungan pulsa sehingga keluaran dari *water flow meter* dapat berupa liter air yang mengalir. Karena sebenarnya keluaran dari *water flow meter* adalah kecepatan air. Program untuk menyimpan penghitungan pulsa dapat dilihat pada Gambar 3.29 dibawah ini.

```
void rpm ()
{
  flow_frequency++;
}
```

Gambar 3.29 Program Penghitungan Pulsa pada Arduino

Setelah pulsa diketahui maka langkah selanjutnya adalah menghitung kecepatan aliran air dan liter air yang telah melewati *water flow meter*. Program untuk menghitung kecepatan aliran air pada *water flow meter* dituliskan pada *void loop* terdapat pada Gambar 3.30 dibawah ini.

```
{
  flow_frequency = 0; // mereset perhitungan
  sei(); //mengaktifkan interrupt
  delay(10);
  cli();
  flowRate = (flow_frequency * 60 / 7.5 / 1900; // Pulse
```

Gambar 3.30 Program Menghitung Kecepatan Aliran Air pada *Flow Meter*

Setelah terhitung berapa kecepatan air yang mengalir, kemudian akan disimpan dalam variabel "*flowRate*". Setelah itu, maka akan diketahui berapa debit air yang mengalir pada sensor melalui program pada Gambar 3.31 dibawah ini.


```
flowMilliLitres += (flowrate) ;  
totalMilliLitres = flowMilliLitres;  
Serial.print.In (totalMilliLitres);
```

Gambar 3.31 Program Menghitung Debit Air yang Mengalir

Program pada Gambar 3.30 berfungsi untuk mencari liter air yang telah melewati *water flow meter*, dan kemudian menyimpannya dalam variabel “*totalMilliLitres*”. Dengan program-program itu, maka pembacaan *water flow meter* dapat dilakukan dan menghasilkan debit air yang mengalir pada sensor *water flow meter*.

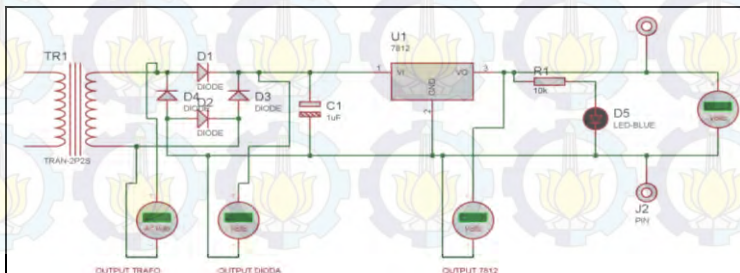
BAB IV HASIL DAN IMPLEMENTASI

Pada bab ini akan dibahas tentang pengujian tiap-tiap rangkaian yang menyusun alat ini. Pengujian dimaksudkan untuk memastikan bahwa kinerja masing-masing alat yang telah dibuat dapat berfungsi sesuai dengan yang diharapkan. Setelah kita merancang dan membuat perangkat keras (*hardware*) dan juga perangkat lunak (*software*), maka untuk mengetahui apakah langkah-langkah yang kita lakukan dan proses yang kita kerjakan sudah mencapai tujuan-tujuan yang diinginkan serta mengetahui apa kekurangan yang bisa dikembangkan lebih lanjut maka dilakukan pengujian dan analisa. Untuk lebih jelasnya pengujian dan analisa ini dibagi menjadi 2 aspek, yaitu aspek *hardware* dan *software*. Secara umum *hardware* merupakan rangkaian-rangkaian yang telah dibuat, dan *software* merupakan program dan *interface* yang dibuat.

Pengujian dari *hardware* akan dilakukan pengukuran dan cara kerja dari rangkaian yang kami buat yang nantinya akan menghasilkan data-data sesuai dengan *hardware* masing-masing. Beberapa *hardware* yang akan diuji adalah *power supply*, minimum sistem ATmega328, *water flow meter*, *solenoid valve*, *relay*, dan *smart card*.

4.1 Pengujian Rangkaian *Power Supply*

Pengujian rangkaian *power supply* ini dilakukan untuk mengetahui keadaan rangkaian tersebut apakah keluaran yang dihasilkan telah sesuai dengan yang diharapkan. Pengujian rangkaian *power supply* terdiri dari keluaran trafo, *output data*, *output 7805*, dan *output adaptor*. Gambar Rangkaian Pengujian *Power Supply* dapat dilihat pada Gambar 4.1 dibawah ini.



Gambar 4.1 Rangkaian Pengujian *Power Supply*

Data hasil pengujian *power supply* ditunjukkan pada Tabel 4.1 dibawah ini.

Tabel 4.1 Tabel Pengujian Rangkaian *Power Supply*

Keluaran Trafo	13V AC
Output Dioda	17V AC
Output 7812	12,5V DC
Output Adaptor	12,5V DC

Pada Tabel 4.1 didapatkan beberapa data setelah melakukan pengukuran pada *power supply* yaitu keluaran trafo sebesar 13V AC, *output data* sebesar 17V AC, *output 7805* sebesar 12,5V DC, dan *output adaptor* sebesar 12,5VDC.

4.2 Pengujian sensor *Water Flow Meter*

Pengujian sensor *water flow meter* dilaksanakan untuk mengetahui berapa pulsa yang didapatkan agar mendapatkan data yang berfungsi sebagai bahan untuk memperbaiki program. Pengujian sensor *water flow meter* ini dilakukan dengan cara membaca keluaran dari sensor menggunakan Arduino dengan menggunakan rumus dari datasheet yaitu:

$$flowrate = hasilsensor \times 60 \div 7.5$$

Pengujian tersebut dapat dilihat pada Tabel 4.2 dibawah ini.

Tabel 4.2 Tabel Pengujian sensor *Water Flow Meter I*

Air (Liter)	<i>Pulse Count</i> pada <i>Flow Meter</i>
1	1876
2	3724
3	7624

Air (Liter)	<i>Pulse Count</i> pada <i>Flow Meter</i>
4	7624
5	9526

Setelah dilakukan pengujian, maka didapatkan rata-rata perubahan *pulsecount* perliternya adalah 1900. Maka rumusnya akan menjadi:

$$flowrate = (hasilsensor \times 60 \div 7.5) \div 1900$$

Pengujian sensor *water flow meter* juga dilakukan ketika program telah dilakukan perbaikan. Pengujian sensor *water flow meter* ini dilakukan dengan tujuan untuk mengetahui berapa digit *counter* yang dikeluarkan ketika air mengalir dalam beberapa liter. Pengujian sensor *water flow meter* ini dilakukan dengan cara mengalirkan air (per liter) dan air tersebut melewati sensor *flow meter*. Pengujian tersebut dapat dilihat pada Tabel 4.3 dibawah ini.

Tabel 4.3 Tabel Pengujian sensor *Water Flow Meter II*

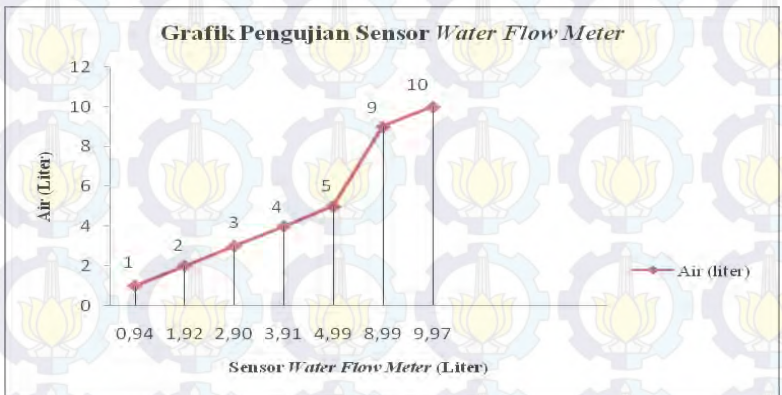
Air (Liter)	Sensor <i>Flow Meter</i> (Liter)	% error
1	0,94	6%
2	1,92	4%
3	2,90	3%
4	3,91	2%
5	4,99	0,1%
9	8,99	0.1%
10	9,97	0,1%

Pada Tabel 4.3 menunjukkan bahwa aliran air (per liter) tidak selalu sama dengan keluaran pada sensor *water flow meter*. Ketika mengalirkan air sebanyak 20 L, sensor *flow meter* menunjukkan angka 21. Begitu juga ketika mengalirkan air sebanyak 25-50 L, keluaran pada

sensor *water flow meter* tidak sesuai dengan aliran air (per liter). Persen *error* pada pengujian sensor *water flow meter* ini, dibuktikan dengan menggunakan rumus :

$$\%error = \frac{\text{Nilai Sebenarnya} - \text{Nilai yang Terbaca}}{\text{Nilai Sebenarnya}} \times 100\%$$

Dengan begitu didapatkan hasil persen *error* dari masing-masing data, agar dapat diketahui ketelitian dari sensor *water flow meter*. Grafik Pengujian Sensor *Water Flow Meter* dapat dilihat pada Gambar 4.2 dibawah ini.



Gambar 4.2 Grafik Pengujian Sensor *Water Flow Meter*

Grafik Pengujian Sensr *Water Flow Meter* diatas menunjukkan hasil pengujian yang ditunjukkan seperti pada Tabel 4.3 dimana persen *error* maksimum yang dihasilkan tidak mencapai angka 10%. Tabel pengujian waktu pada sensor *water flow meter* dapat dilihat pada Tabel 4.4 dibawah ini.

Tabel 4.4 Tabel Pengujian Waktu pada Sensor *Water Flow Meter*

Liter / detik	Hz
0.077	37.6
0.09	43.9

Liter / detik	Hz
0.088	43.1
0.089	43.5
0.09	43.9
0.102	49.9
0.107	52.3
0.11	53.8
0.108	52.8
0.114	55.7
0.113	55.2
0.108	52.8

Pada Tabel 4.4 menunjukkan bahwa pengujian pada sensor *water flow meter*, untuk mencari frekuensi yang dihasilkan. Dengan cara mengalirkan air sesuai dengan liter yang ditentukan dan menghitung waktunya agar mendapatkan data dengan satuan liter per detik. Untuk menghitung frekuensi dihasilkan menggunakan alat ukur osiloskop atau avometer digital khusus yang dapat menghitung frekuensi. Cara menghubungkannya, kaki positif dari sensor *water flow meter* dihubungkan dengan VCC 5V dan kaki GND dihubungkan dengan GND (sumber). Kaki positif dari alat ukur dihubungkan ke output sensor dan kaki negatif alat ukur dihubungkan ke GND (sumber).

4.3 Pengujian Smart Card

Pengujian *Smart Card* dilakukan untuk mengetahui keluaran yang dihasilkan oleh *Smart Card* tersebut. Pengujian terhadap *Smart Card* dilakukan dengan memasukkan data pada *Smart Card* (RFID Mifare 522) terlebih dahulu kemudian dilakukan pembacaan oleh mikrokontroler dan ditampilkan pada LCD 20x4. Dari pengujian yang

telah dilakukan, maka didapatkan data seperti terlihat pada Tabel 4.5 di bawah ini.

Tabel 4.5 Pengujian Data yang Terbaca pada *Smart Card*

Data yang Diisi pada <i>Smart Card</i> (Desimal)	Data yang Terbaca dari <i>Smart Card</i> (Desimal)
255	255
200	200

Pengujian pengambilan data pada *smart card* dilakukan untuk memastikan agar dapat mengetahui berapa *error* kesalahan yang terjadi. Pengujian pengambilan data pada *smart card* dilakukan dengan cara mengambil *byte* data yang digunakan lalu dikurangi dengan nilai yang telah ditentukan. Nilai yang telah ditentukan adalah angka 5. Hasil dari pengujian pengambilan data pada *smart card* dapat dilihat pada Tabel 4.6 dibawah ini.

Tabel 4.6 Tabel Pengujian Pengambilan Data pada *Smart Card*

Isi Awal	Hasil Setelah Diproses oleh Program
255	250
250	245
245	240
240	235
20	15
15	10
10	5
5	0

Isi Awal	Hasil Setelah Diproses oleh Program
4	4
3	3
2	2
1	1

Apabila isi awal pada *smart card* diatas angka 5 maka akan terjadi proses pengurangan yang dilakukan program dan penulisan hasil dari pengurangan tersebut. Apabila nilai yang awal kurang dari angka 5 maka tidak terjadi proses pengurangan dan penulisan. *Error* yang terjadi pada pengujian pengambilan data pada *smart card* sebesar 0%.

Pengujian pada *smart card* juga dilakukan dengan melakukan pengukuran jarak pembacaan *smart card* RFID Mifare RC522 agar dapat mengetahui berapa jarak minimum dan maksimum antara RC522 dengan RFID *tag*. Pengujian ini dilakukan dengan cara melakukan pengukuran menggunakan alat bantu penggaris. Data yang didapatkan dapat dilihat pada Tabel 4.7 dibawah ini.

Tabel 4.7 Tabel Pengukuran Jarak Pembacaan *Smart Card* RFID Mifare RC522

Jarak <i>Smart Card</i>	Pembacaan <i>Smart Card</i>	
	Tanpa Ada Halangan	Terhalang oleh Akrilik
5 cm	Tidak Terbaca	Tidak Terbaca
4 cm	Tidak Terbaca	Tidak Terbaca
3,5 cm	Terbaca	Terbaca
3 cm	Terbaca	Terbaca
2 cm	Terbaca	Terbaca
1 cm	Terbaca	Terbaca
0 cm	Terbaca	-

Jarak minimum penggunaan antara RC522 dengan RFID *tag* sebesar 0 cm dan jarak maksimum sebesar 3,5 cm. Tidak terjadi perbedaan data

antara jarak minimum dan maksimum, bahkan ketika terhalang oleh akrilik jarak maksimumnya dan data yang terbaca tetap.

4.4 Pengujian Alat

Pengujian Alat dilakukan untuk mengetahui apakah alat sudah berfungsi sebagaimana mestinya secara keseluruhan. Baik LCD, *Buzzer*, *solenoid valve*, dan sensor *water flow meter*. Pengujian alat dilakukan dengan cara mengisi smartcard terlebih dahulu pada alat, lalu air akan dialirkan pada alat sebagaimana fungsi alat yaitu sebagai alat penghitungan dan pembayaran air. Pembacaan hasil sisa liter air akan ditampilkan oleh LCD 20x4. Selain itu, terdapat indikator penanda jika hasil sisa liter air telah memasuki nilai 25 Liter, maka *buzzer* akan menyala. Nilai 25 sendiri telah ditentukan pada program. Dari pengujian yang telah dilakukan, maka didapatkan data seperti terlihat pada Tabel 4.8 dibawah ini.

Tabel 4.8 Tabel Pengujian Alat Secara keseluruhan

Sisa Liter	Buzzer	Valve
50	Off	On
47	Off	On
41	Off	On
36	Off	On
32	Off	On
25	On	On
23	On	On
12	On	On
9	On	On
4	On	On
0	On	Off

Jadi, ketika sisa liter air pada alat melebihi 25, maka air dapat mengalir tanpa ada bunyi *buzzer*. Ketika sisa liter air mencapai 25, air tetap bisa mengalir tetapi *buzzer* akan berbunyi sebagai penanda karena sisa air

yang bisa digunakan tinggal 25 Liter. *Buzzer* akan terus menyala sampai sisa liter air ditambah menggunakan smartcard. Ketika sisa liter air telah mencapai nilai 1, maka *solenoid valve* akan *off* dan air tidak dapat lagi mengalir. Selain itu, *buzzer* akan terus menyala. Air dapat mengalir lagi jika sisa liter air ditambah menggunakan smartcard.

Pengujian *smart card* pada alat dilakukan untuk mengetahui bagaimana respon alat ketika *smart card* dimasukkan ke *reader*. Pengujian *smart card* pada alat dilakukan dengan cara memasukkan *smart card* pada tempat yang telah disediakan saat air masih mengalir pada sensor. Dari pengujian yang telah dilakukan, maka didapatkan data seperti terlihat pada Tabel 4.9 dibawah ini.

Tabel 4.9 Tabel Pengujian Alat secara keseluruhan

Kartu	Buzzer	Valve	Air
Ada	Off	Off	Tidak Mengalir
Tidak Ada	Off	On	Mengalir

Jika pada alat tidak terdapat kartu, maka air tetap bisa mengalir. Ketika kartu dimasukkan pada alat, *solenoid valve* akan mati (menutup) dan air tidak akan mengalir. *Buzzer* tidak berpengaruh pada pengujian ini.



BAB V

PENUTUP

Dari hasil pengujian yang telah dilakukan dengan menggunakan beberapa alat bantu sebagai perbandingan, dapat diambil kesimpulan bahwa sensor *water flow meter* digunakan untuk menghitung debit air yang memiliki persentase *error* maksimum sebesar 6%. Dan pembayaran PDAM menggunakan *smart card* memiliki persentase *error* sebesar 0%. Selain itu, pemilihan solenoid valve dirasa kurang tepat karena solenoid valve yang digunakan hanya bisa mengalirkan air dengan tekanan minimal 0.02 Mpa.

Saran untuk penelitian selanjutnya adalah menggunakan sensor yang digunakan untuk menghitung debit air yang lebih baik agar persentase *error* lebih sedikit dan disarankan menggunakan *solenoid valve* yang dapat mengalirkan air tanpa tekanan minimal (tekanan minimal sebesar 0 Mpa).



DAFTAR PUSTAKA

- [1] Inviro, 2015. “Fungsi dan Peran Air Bagi Kehidupan Manusia”. <http://training.inviro.co.id/fungsi-dan-peran-air-bagi-kehidupan-manusia/>
- [2] Hendri Iksan, 2015. “Arsitektur Mikrokontroler”. <http://kl601.ilearning.me/2015/10/17/essay-10-arsitektur-mikrocontroller-2/>
- [3] Addi Core, 2012. “RFID RC522 - RFID *Quick Start Guide*”. https://www.addicore.com/v/vspfiles/downloadables/Product%20Downloadables/RFID_RC522/RFIDQuickStartGuide.pdf
- [4] Agus Purnama, 2012. “Pengertian dan Komponen *Radio Frequency Identification*”. <http://elektronika-dasar.web.id/pengertian-dan-komponen-radio-frequency-identification-rfid/>
- [5] Agus Purnama, 2016. “Alokasi Frekuensi Kerja RFID (*Radio Frequency Identification*)”. <http://elektronika-dasar.web.id/alokasi-frekuensi-kerja-rfid-radio-frequency-identification/>
- [6] Agus Purnama, 2012. “LCD (*Liquid Cristal Display*)”. <http://elektronika-dasar.web.id/lcd-liquid-cristal-display/>
- [7] Amir Hamzah, 2013. “LCD 20 X 4”. <https://www.scribd.com/doc/185920131/LCD-20X4>
- [8] Rossdeco, 2012. “Fungsi *Solenoid Valve*”. <http://dokumen.tips/documents/fungsi-solenoid-valve.html>
- [9] Rangkaian Elektronika, 2016. “Rangkaian *Buzzer*”. <http://www.rangkaianelektronika.org/rangkaian-buzzer.htm>
- [10] Rudy Wiratama, 2012. “*Water Flow Meter*”. <https://rudywinoto.com/2012/02/27/461/>
- [11] Wikipedia, 2015. “Arduino”. <https://id.wikipedia.org/wiki/Arduino>
- [12] Teknik Elektronika, 2015. “Pengertian dan Fungsi *Relay*”. <http://teknikelektronika.com/pengertian-relay-fungsi-relay/>
- [13] Teknik Elektronika, 2015. “Prinsip Kerja DC *Power Supply*”

(Adaptor)". <http://teknikelektronika.com/prinsip-kerja-dc-power-supply-adaptor/>

[14] Subekti Pranoto, ST., 2008. "Klasifikasi Pelanggan dan Tarif Air Minum PDAM Kota Surabaya". http://www.pdam-sby.go.id/page.php?get=tampil_tabel_tarif&bhs=1

[15] Nanang Fakhur Rozi, "*Aplikasi RFID Untuk Pembayaran TOL*", PENS-ITS, 2008.

LAMPIRAN A

```
#include <SPI.h>
#include <MFRC522.h>
#include <LiquidCrystal.h>
#define RST_PIN      9           // Configurable, see typical pin layout
above
#define SS_PIN       10          // Configurable, see typical pin layout
above
LiquidCrystal lcd(8, 3, 4, 5, 6, 7);
MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance.

MFRC522::MIFARE_Key key;

byte sensorInterrupt = 0; // 0 = pin 2; 1 = pin 3
byte sensorPin       = 2;
float calibrationFactor = 4.5;
volatile byte pulseCount;
float flowRate; //deklarasi flowrate
float flowMilliLitres;
float totalMilliLitres;
float kapasitas;
unsigned long oldTime;
int x;
int y=5;
int z;
int limit = A1;
int relay = A2;
int buzzer = A0;

void setup() {
  pinMode(A1,INPUT);
  pinMode(A2,OUTPUT);
  pinMode(A0,OUTPUT);
  lcd.begin(20, 4);
  Serial.begin(9600);
  while (!Serial);
  SPI.begin();
  mfrc522.PCD_Init();
```

```

for (byte i = 0; i < 6; i++) {
    key.keyByte[i] = 0xFF;
}
lcd.setCursor(4,0);
lcd.print("Tugas Akhir");
delay(500);
lcd.setCursor(2,1);
lcd.print("Penghitungan dan");
lcd.setCursor(2,2);
lcd.print("Pembayaran Aliran");
lcd.setCursor(8,3);
lcd.print("Air");
delay(1500);
lcd.clear();
pulseCount    = 0;
flowRate      = 0.0;
flowMilliLitres = 0;
totalMilliLitres = 0;
oldTime       = 0;
attachInterrupt(sensorInterrupt, pulseCounter, FALLING);
}

void loop() {
    if((millis() - oldTime) > 1000){
        detachInterrupt(sensorInterrupt);
        flowRate = (((1000.0 / (millis() - oldTime)) * pulseCount) /
calibrationFactor;
        oldTime = millis();
        flowMilliLitres = (float)(flowRate / 60) * 1000;
        totalMilliLitres += flowMilliLitres;
        unsigned int frac;
        frac = (flowRate - int(flowRate)) * 10;
        lcd.setCursor(0, 0);
        lcd.print("Flow:");
        if(int(flowRate) < 10)
        {
            lcd.print(" ");
        }
    }
}

```

```

lcd.print((int)flowRate); // Print the integer part of the variable
lcd.print('.');           // Print the decimal point
lcd.print(frac, DEC);     // Print the fractional part of the variable
lcd.print("L/Min");
lcd.setCursor(8, 1);
lcd.print("L=");
lcd.print(totalMilliLitres / 1000);
Serial.println("A");Serial.println(flowMilliLitres/1000);
Serial.println("B");Serial.println(totalMilliLitres / 1000);
pulseCount = 0;
attachInterrupt(sensorInterrupt, pulseCounter, FALLING);
delay(1000);
//if 1
if (digitalRead(limit) == LOW)
{
    lcd.setCursor(0, 2);
    lcd.print("          ");
    lcd.setCursor(0, 3);
    lcd.print("Tidak Mendeteksi Kartu");
    Serial.println("OFF");
}
//if 1
if (digitalRead(limit) == HIGH)
{
    digitalWrite(relay, LOW);
    Serial.println("ON");
    lcd.setCursor(0, 3);
    lcd.print("          ");
    lcd.setCursor(0, 2);
    lcd.print("          ");
    lcd.setCursor(0, 2);
    lcd.print("membaca kartu");
    delay (1000);
    //if 1a
    if ( ! mfrc522.PICC_IsNewCardPresent())
        return;
    //if 1a
    if ( ! mfrc522.PICC_ReadCardSerial())
        return;
}

```

```

lcd.setCursor(0, 2);
lcd.print("          ");
delay (1000);
lcd.setCursor(0, 2);
lcd.print("type Kartu");
MFRC522::PICC_Type piccType =
mfrc522.PICC_GetType(mfrc522.uid.sak);
lcd.setCursor(10, 2);
lcd.print(mfrc522.PICC_GetTypeName(piccType));
Serial.println(mfrc522.PICC_GetTypeName(piccType));
delay (1000);
//if 1b
if ( piccType != MFRC522::PICC_TYPE_MIFARE_MINI
    && piccType != MFRC522::PICC_TYPE_MIFARE_1K
    && piccType != MFRC522::PICC_TYPE_MIFARE_4K) {
    return;
}
byte sector = 1;
byte blockAddr = 4;
byte dataBlock[] = {
    0x01, 0xff, 0xff, 0xff, // 1, 2, 3, 4,
    0xff, 0xff, 0xff, 0xff, // 5, 6, 7, 8,
    0xff, 0xff, 0xff, 0xff, // 9, 10, 255, 12,
    0xff, 0xff, 0xff, 0xff // 13, 14, 15, 16
};
byte trailerBlock = 7;
MFRC522::StatusCode status;
byte buffer[18];
byte size = sizeof(buffer);
status = (MFRC522::StatusCode)
mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY
_A, trailerBlock, &key, &(mfrc522.uid));
if (status != MFRC522::STATUS_OK) {
    lcd.setCursor(8, 2);
    lcd.print("failed: ");
    Serial.println(mfrc522.GetStatusCodeName(status));
    return;
}
delay (100);

```



```

// Read data from the block
lcd.setCursor(0, 2);
lcd.print("      ");
lcd.setCursor(0, 2);
lcd.print("Membaca:");
lcd.setCursor(0, 3);
lcd.print("Data pada"); Serial.print(blockAddr);
lcd.setCursor(9, 3);
lcd.print("byte0=");
status = (MFRC522::StatusCode)
mfrc522.MIFARE_Read(blockAddr, buffer, &size);
x=buffer[0];
lcd.setCursor(15, 3);
lcd.print(x);
if (status != MFRC522::STATUS_OK) {
    lcd.setCursor(8,3);
    lcd.print("failed: ");
    Serial.println(mfrc522.GetStatusCodeName(status));
}
delay (2000);
if ( x <= 5 )
{
    lcd.setCursor(0,3);
    lcd.print("kartu anda kurang");
    delay (3000);
    lcd.clear();
    delay (1000);
    Serial.println("no");
}

if ( x >= 5 ) {
    Serial.println("yes");
    lcd.setCursor(0, 2);
    lcd.print("      ");
    lcd.setCursor(0, 3);
    lcd.print("      ");
    lcd.setCursor(0, 2);
    lcd.print("Terbaca");
    lcd.setCursor(9, 2);

```

```

lcd.print(buffer [0]);
lcd.setCursor(12, 2);
lcd.print("0 LITER");
dataBlock[0]= x - y;
lcd.setCursor(0, 3);
lcd.print("dikurangi 50L");
kapasitas=kapasitas+(y*10);
delay (3000);
Serial.println(F("Authenticating again using key B..."));
status = (MFRC522::StatusCode)
mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY
_B, trailerBlock, &key, &(mfrc522.uid));
if (status != MFRC522::STATUS_OK) {
    lcd.setCursor(0,1);
    lcd.print("PCD_Authenticate() failed: ");
    Serial.println(mfrc522.GetStatusCodeName(status));
    return;
}
// Write data to the block
Serial.print(blockAddr);
dump_byte_array(dataBlock, 16); Serial.println();
status = (MFRC522::StatusCode)
mfrc522.MIFARE_Write(blockAddr, dataBlock, 16);
if (status != MFRC522::STATUS_OK) {
    lcd.setCursor(0,2);
    lcd.print("MIFARE_Write() failed: ");
    Serial.println(mfrc522.GetStatusCodeName(status));
}
delay (50);
lcd.setCursor(0, 2);
lcd.print("    ");
lcd.setCursor(0,2);
lcd.print("Sisa Liter:");
z=(x-y);
lcd.print(z);
lcd.print("0 LITER");
delay (1000);
lcd.setCursor(0, 2);
lcd.print("    ");

```

```

    delay (1000);
    lcd.setCursor(0, 3);
    lcd.print("        ");
    lcd.setCursor(0, 3);
    lcd.print("Silahkan Ambil Kartu");
    delay (3000);

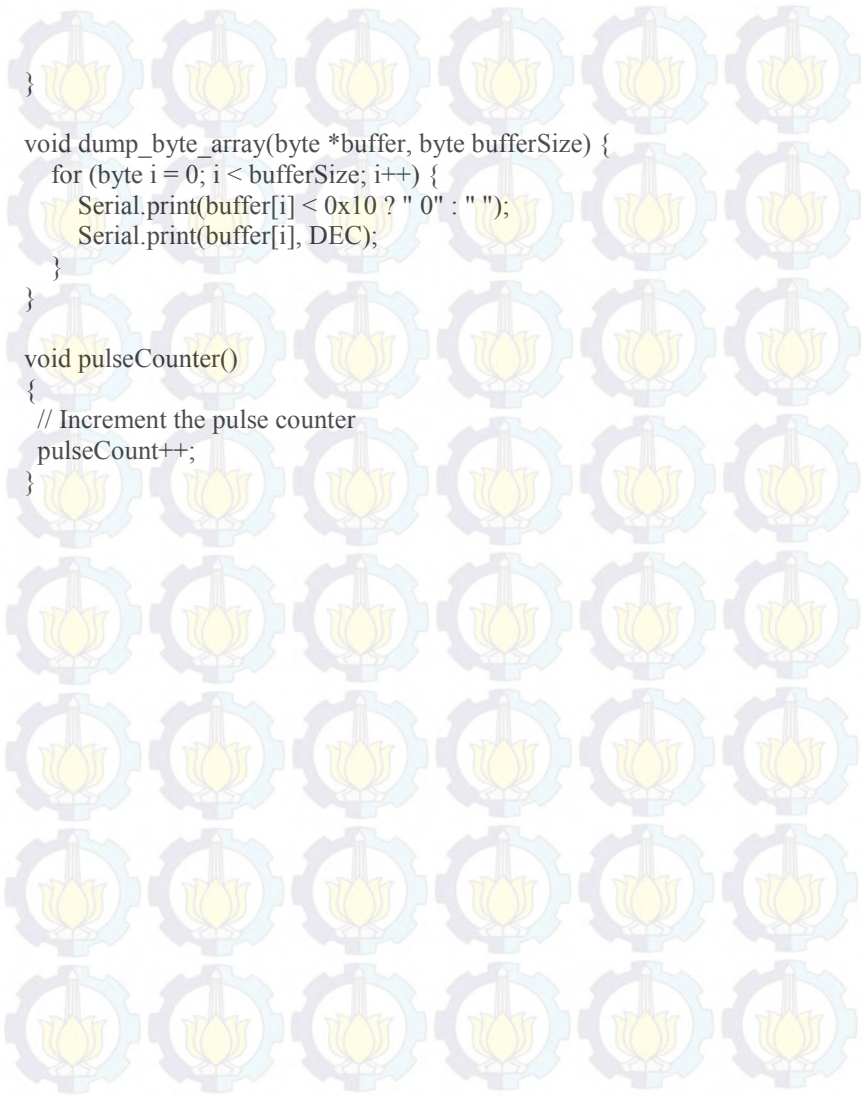
    // Halt PICC
    mfrc522.PICC_HaltA();
    // Stop encryption on PCD
    mfrc522.PCD_StopCrypto1();

    return;
}

}

kapasitas ==(flowMilliLitres/1000);
Serial.println("C");Serial.println(kapasitas);
delay(1000);
lcd.setCursor(0, 1);
lcd.print("        ");
lcd.setCursor(0, 1);
lcd.print(kapasitas);
delay (100);
if (kapasitas <= 25)
{
    digitalWrite(buzzer,LOW);
    digitalWrite(relay,HIGH);
}
if (kapasitas <=1)
{
    digitalWrite(relay,LOW);
}
if (kapasitas >=26)
{
    digitalWrite(buzzer,HIGH);
    digitalWrite(relay,HIGH);
}
}

```



```
}  
  
void dump_byte_array(byte *buffer, byte bufferSize) {  
    for (byte i = 0; i < bufferSize; i++) {  
        Serial.print(buffer[i] < 0x10 ? " 0" : " ");  
        Serial.print(buffer[i], DEC);  
    }  
}  
  
void pulseCounter()  
{  
    // Increment the pulse counter  
    pulseCount++;  
}
```


Data Sheet ATmega328

Features

- High-performance, Low-power AVR[®] 8-bit Microcontroller
- Advanced RISC Architecture
 - 131 Powerful Instructions – Most Single-clock Cycle Execution
 - 32 x 8 General Purpose Working Registers
 - Fully Static Operation
 - Up to 16 MIPS Throughput at 16 MHz
 - On-chip 2-cycle Multiplier
- Nonvolatile Program and Data Memories
 - 32K Bytes of In-System Self-Programmable Flash
 - Endurance: 10,000 Write/Erase Cycles
 - Optional Boot Code Section with Independent Lock Bits
 - In-System Programming by On-chip Boot Program
 - True Read-While-Write Operation
 - 1024 Bytes EEPROM
 - Endurance: 100,000 Write/Erase Cycles
 - 2K Byte Internal SRAM
 - Programming Lock for Software Security
- JTAG (IEEE std. 1149.1 Compliant) Interface
 - Boundary-scan Capabilities According to the JTAG Standard
 - Extensive On-chip Debug Support
 - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
 - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
 - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
 - Real Time Counter with Separate Oscillator
 - Four PWM Channels
 - 8-channel, 10-bit ADC
 - 8 Single-ended Channels
 - 7 Differential Channels in TQFP Package Only
 - 2 Differential Channels with Programmable Gain at 1x, 10x, or 200x
 - Byte-oriented Two-wire Serial Interface
 - Programmable Serial USART
 - Master/Slave SPI Serial Interface
 - Programmable Watchdog Timer with Separate On-chip Oscillator
 - On-chip Analog Comparator
- Special Microcontroller Features
 - Power-on Reset and Programmable Brown-out Detection
 - Internal Calibrated RC Oscillator
 - External and Internal Interrupt Sources
 - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
- I/O and Packages
 - 32 Programmable I/O Lines
 - 40-pin PDIP, 44-lead TQFP, and 44-pad QFN/MLF
- Operating Voltages
 - 2.7 - 5.5V for ATmega32L
 - 4.5 - 5.5V for ATmega32
- Speed Grades
 - 0 - 8 MHz for ATmega32L
 - 0 - 16 MHz for ATmega32
- Power Consumption at 1 MHz, 3V, 25°C for ATmega32L
 - Active: 1.1 mA
 - Idle Mode: 0.35 mA
 - Power-down Mode: < 1 µA



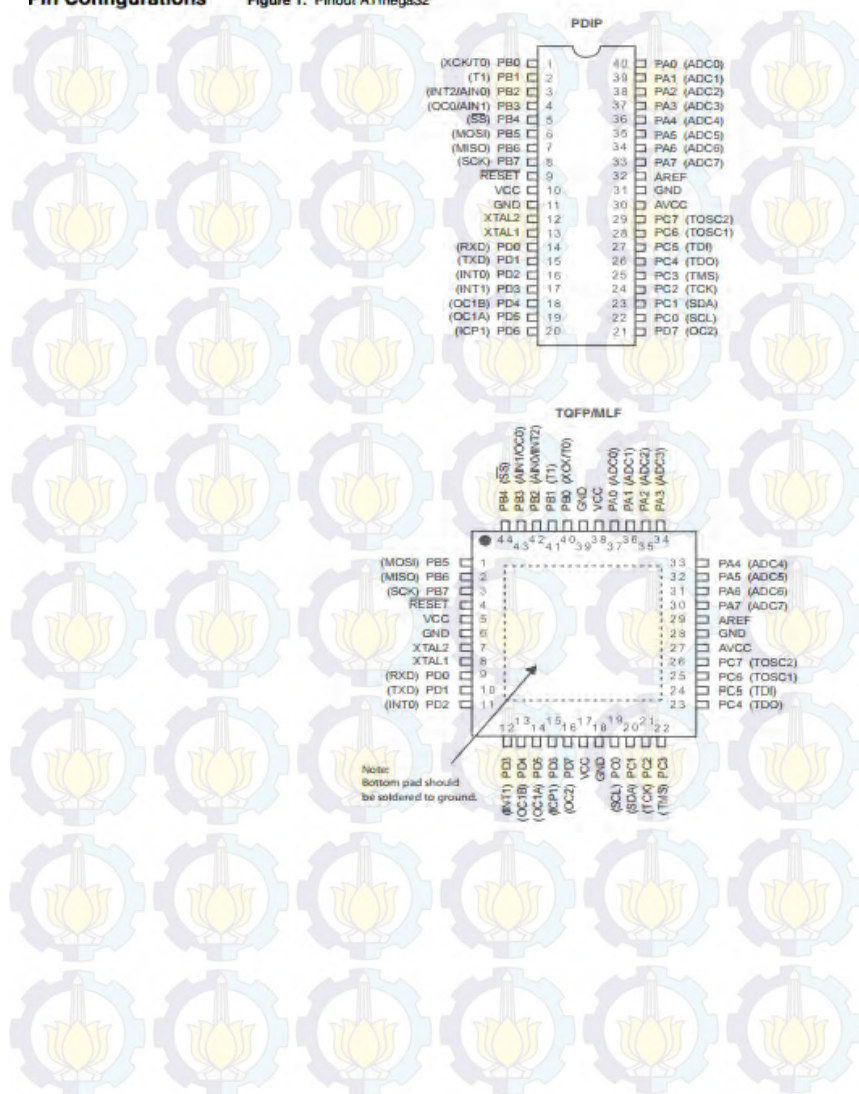
8-bit AVR[®]
Microcontroller
with 32K Bytes
In-System
Programmable
Flash

ATmega32
ATmega32L

Summary

Pin Configurations

Figure 1. Pinout ATmega32



Block Diagram

Figure 2. Block Diagram

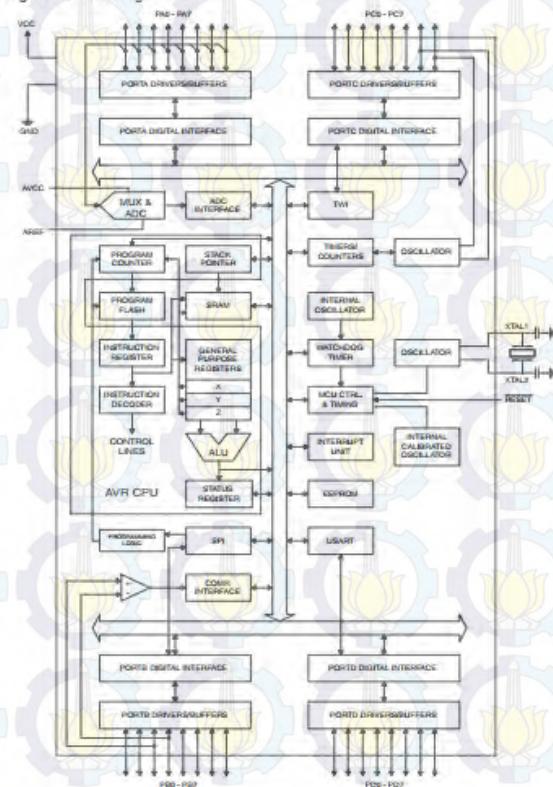
Pin Descriptions

VCC
Digital supply voltage.

GND
Ground.

Port A (PA7..PA0)
Port A serves as the analog inputs to the A/D Converter.
Port A also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Figure 2. Block Diagram



Pin Descriptions

VCC

Digital supply voltage.

GND

Ground.

Port A (PA7..PA0)

Port A serves as the analog inputs to the A/D Converter.

Port A also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B (PB7..PB0)

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B also serves the functions of various special features of the ATmega32 as listed on page 57.

Port C (PC7..PC0)

Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs.

The TD0 pin is tri-stated unless TAP states that shift out data are entered.

Port C also serves the functions of the JTAG interface and other special features of the ATmega32 as listed on page 60.

Port D (PD7..PD0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port D also serves the functions of various special features of the ATmega32 as listed on page 62.

RESET

Reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in Table 15 on page 37. Shorter pulses are not guaranteed to generate a reset.

XTAL1

Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting Oscillator amplifier.

AVCC

AVCC is the supply voltage pin for Port A and the A/D Converter. It should be externally connected to V_{CC} , even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter.

AREF

AREF is the analog reference pin for the A/D Converter.

Resources

A comprehensive set of development tools, application notes and datasheets are available for download on <http://www.atmel.com/avr>.

Data Sheet Sensor Water Flow Meter

YIFA the plastics Ltd Product Introduction

1. Model: YF-21
2. Product Name: Hall sensor
3. Flow Range: 1-30L/MIN
4. (1) Connection Method



(2) Voltage Range 3.5-24VDC, Pulse Characteristic: $f=7Q(L/MIN)$.

(3) Extent of error: $\pm 5\%$.

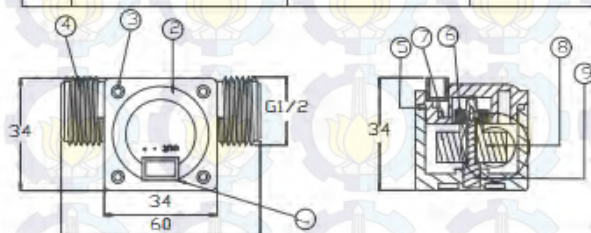
(4) Flow-Pulse

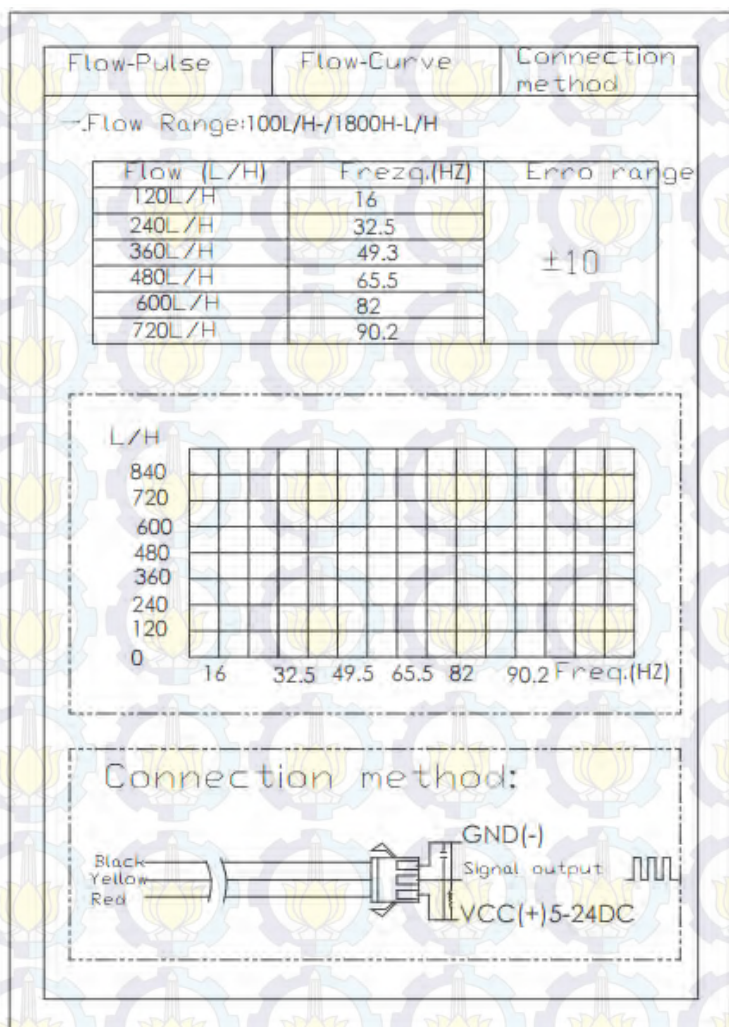
2L/MIN=16HZ 4L/MIN=32.5HZ 6L/MIN=49.3HZ

8L/MIN=65.5HZ 10L/MIN=82HZ

5. BOM

No.	Item	Material	Qty.
1	Connection wire		1
2	Bonnet	PA	1
3	Screw		4
4	Valve body	PA	1
5	Leak press. valve		1
6	Magnet		1
7	Hall		1
8	Impeller	POM	1
9	Rustless steel axis	SUS304	1
10			
11			





Data Sheet MFRC522

4. Quick reference data

Table 1. Quick reference data

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
AV_{DD}	Supply Voltage	$AV_{SS} = DV_{SS} = PV_{SS} = TV_{SS} = 0\text{ V}$, [1] $PV_{DD} \leq AV_{DD} = DV_{DD} = TV_{DD}$ [1]	2.5	-	3.6	V
DV_{DD}						
TV_{DD}						
PV_{DD}	Pad power supply	$AV_{SS} = DV_{SS} = PV_{SS} = TV_{SS} = 0\text{ V}$, [3] $PV_{DD} \leq AV_{DD} = DV_{DD} = TV_{DD}$	1.8	-	3.6	V
SV_{DD}	MF1MF0UT Pad Power Supply	$AV_{SS} = DV_{SS} = PV_{SS} = TV_{SS} = 0\text{ V}$, [3]	1.8	-	3.6	V
I_{HPD}	Hard Power-down Current	$AV_{DD} = DV_{DD} = TV_{DD} = PV_{DD} = 3\text{ V}$, [4] $N_{RESET} = \text{LOW}$	-	-	5	μA
I_{SPD}	Soft Power-down Current	$AV_{DD} = DV_{DD} = TV_{DD} = PV_{DD} = 3\text{ V}$, [4] RF level detector on	-	-	10	μA
I_{VDD}	Digital Supply Current	$DV_{DD} = 3\text{ V}$	-	6.5	9	mA
I_{VDD}	Analog Supply Current	$AV_{DD} = 3\text{ V}$, bit RCVOFF = 0	-	7	10	mA
$I_{VDD,RCVOFF}$	Analog Supply Current, receiver switched off	$AV_{DD} = 3\text{ V}$, bit RCVOFF = 1	-	3	5	mA
I_{VDD}	Pad Supply Current	[5]	-	-	40	mA
I_{VDD}	Transmitter Supply Current	Continuous Wave [6]	-	60	100	mA
T_{amb}	operating ambient temperature		-25		+85	$^{\circ}\text{C}$

[1] Supply voltage below 3 V reduces the performance (e.g. the achievable operating distance).

[2] AV_{DD} , DV_{DD} and TV_{DD} shall always be on the same voltage level.

[3] PV_{DD} shall always be on the same or lower voltage level than DV_{DD} .

[4] I_{VDD} depends on TV_{DD} and the external circuitry connected to Tx1 and Tx2

[5] I_{VDD} depends on the overall load at the digital pins.

[6] During operation with a typical circuitry the overall current is below 100 mA.

[7] I_{SPD} and I_{HPD} are the total currents over all supplies.

[8] Typical value using a complementary driver configuration and an antenna matched to 40 Ω between TX1 and TX2 at 13.56 MHz

6. Block diagram

The Analog interface handles the modulation and demodulation of the analog signals.

The contactless UART handles the protocol requirements for the communication schemes in co-operation with the host. The comfortable FIFO buffer allows a fast and convenient data transfer from the host to the contactless UART and vice versa.

Various host interfaces are implemented to fulfil different customer requirements.

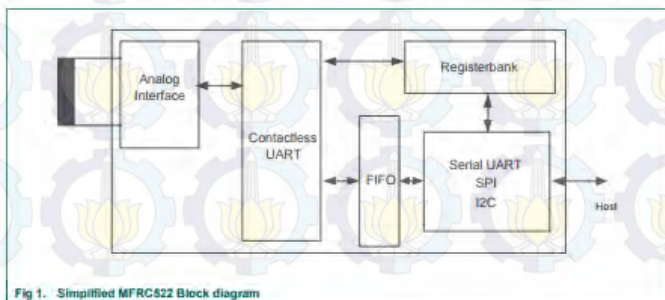
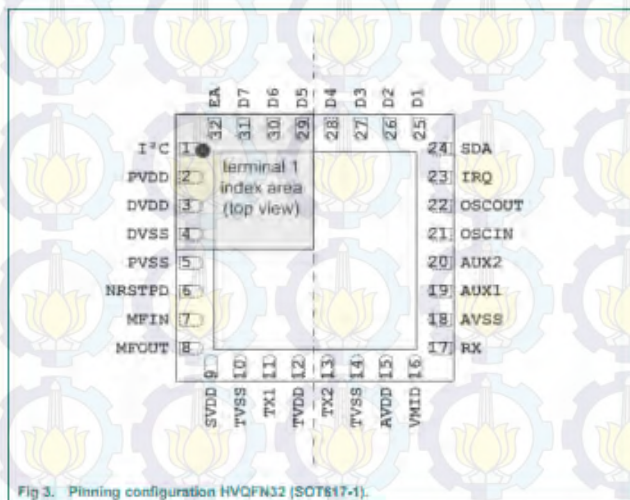


Fig 1. Simplified MFRC522 Block diagram

7.1 Pinning



7.2 Pin description

Table 3: Pin description

Symbol	Pin	Type	Description
PC	1	I	I2C enable
PVDD	2	PWR	Pad power supply
DVDD	3	PWR	Digital Power Supply
DVSS	4	PWR	Digital Ground
PVSS	5	PWR	Pad power supply ground
NRSTPD	6	I	Not Reset and Power-down: When LOW, internal current sinks are switched off, the oscillator is inhibited, and the input pads are disconnected from the outside world. With a positive edge on this pin the internal reset phase starts.
MFIN	7	I	Mifare Signal Input
MFOUT	8	O	Mifare Signal Output
SVDD	9	PWR	MFIN / MFOUT Pad Power Supply: provides power to for the MFIN / MFOUT pads
TVSS	10, 14	PWR	Transmitter Ground: supplies the output stage of TX1 and TX2

Table 3: Pin description ...continued

Symbol	Pin	Type	Description
TX1	11	O	Transmitter 1: delivers the modulated 13.56 MHz energy carrier
TVDD	12	PWR	Transmitter Power Supply: supplies the output stage of TX1 and TX2
TX2	13	O	Transmitter 2: delivers the modulated 13.56 MHz energy carrier
TVSS	10, 14	PWR	Transmitter Ground: supplies the output stage of TX1 and TX2
AVDD	15	PWR	Analog Power Supply
VMID	16	PWR	Internal Reference Voltage: This pin delivers the internal reference voltage.
RX	17	I	Receiver Input. Pin for the received RF signal.
AVSS	18	PWR	Analog Ground
AUX1	19	O	Auxiliary Outputs: These pins are used for testing.
AUX2	20	O	
OSCIN	21	I	Crystal Oscillator Input: input to the inverting amplifier of the oscillator. This pin is also the input for an externally generated clock ($f_{osc} = 27.12$ MHz).
OSCOU	22	O	Crystal Oscillator Output: Output of the inverting amplifier of the oscillator.
IRQ	23	O	Interrupt Request: output to signal an interrupt event
SDA	24	I	Serial Data Line ^[1]
D1	25	I/O	Data Pins for different interfaces (test port, IIC, SPI, UART) ^[2]
D2	26	I/O	
D3	27	I/O	
D4	28	I/O	
D5	29	I/O	
D6	30	I/O	
D7	31	I/O	
EA	32	I	External Address: This Pin is used for coding I2C Address ^[2]

[1] Connection of heat sink pad on package bottom side is not necessary. Optional connection to DVSS is possible.

[2] The pin functionality for the interfaces is explained in [Section 10 "DIGITAL Interfaces"](#).

8. Functional description

MFRC522 transmission module supports the Reader/Writer mode for ISO/IEC 14443A/MIFARE[®] with different transfer speeds and modulation schemes.

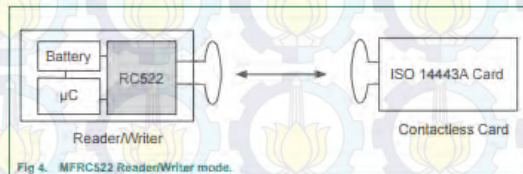
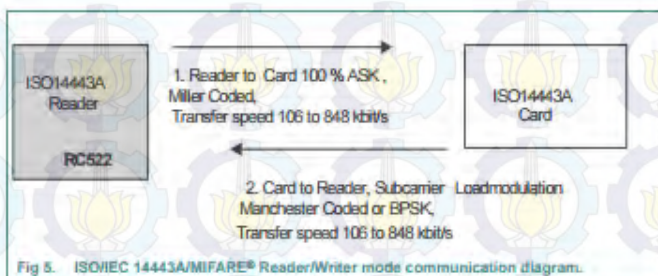


Fig 4. MFRC522 Reader/Writer mode.



The communication overview in [Table 4 "Communication overview for ISO/IEC 14443A/MIFARE® reader/writer"](#) describes the physical parameters.

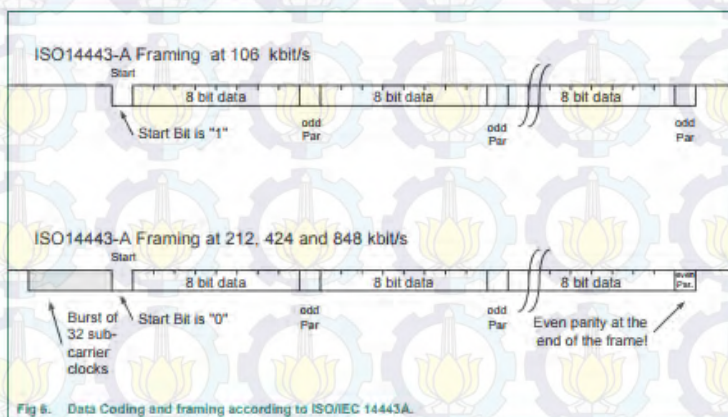
Table 4: Communication overview for ISO/IEC 14443A/MIFARE® reader/writer

Communication direction	ISO/IEC 14443A/ MIFARE®				
	transfer speed	106 kbit/s	212 kbit/s	424 kbit/s	848 kbit/s
Reader → Card (send data from the MFRC522 to a card)	Modulation on reader side	100% ASK	100% ASK	100% ASK	100% ASK
	bit coding	Modified Miller coding	Modified Miller coding	Modified Miller coding	Modified Miller coding
	Bitlength	(128/13.56) μs	(64/13.56) μs	(32/13.56) μs	(16/13.56) μs

Table 4: Communication overview for ISO/IEC 14443A/MIFARE® reader/writer ...continued

Communication direction		ISO/IEC 14443A/ MIFARE®			
	transfer speed	106 kbit/s	212 kbit/s	424 kbit/s	848 kbit/s
Card → Reader (MFRC522 receives data from a card)	modulation on card side	subcarrier load modulation	subcarrier load modulation	subcarrier load modulation	subcarrier load modulation
	subcarrier frequency	13.56 MHz/16	13.56MHz/16	13.56MHz/16	13.56MHz/16
	bit coding	Manchester coding	BPSK	BPSK	BPSK

The contactless UART of MFRC522 and a dedicated external host are required to handle the complete MIFARE® / ISO/IEC 14443A / MIFARE® protocol. The following Figure 6: "Data Coding and framing according to ISO/IEC 14443A." shows the Data Coding and framing according to ISO/IEC 14443A / MIFARE®.



The internal CRC co-processor calculates the CRC value according to the definitions given in the ISO/IEC 14443A part 3 and handles parity generation internally according to the transfer speed. Automatic parity generation can be switched off by bit *ParityDisable* in register *0x1D ManualRCVReg*.

9.1.1 Register Bit Behavior

Depending on the functionality of a register, the access conditions to the register can vary. In principle bits with same behavior are grouped in common registers. In [Table 6](#) the access conditions are described.

Table 6: Behavior of Register Bits and its Designation

Abbreviation	Behavior	Description
r/w	read and write	These bits can be written and read by the μ -Controller. Since they are used only for control means, their content is not influenced by internal state machines, e.g. the <i>Com/EnReg</i> -Register may be written and read by the μ -Controller. It will also be read by internal state machines, but never changed by them.
dy	dynamic	These bits can be written and read by the μ -Controller. Nevertheless, they may also be written automatically by internal state machines, e.g. the <i>CRCReady</i> bit can not be written from external but shows internal states.
r	read only	These register bits hold values which are determined by internal states only, e.g. the <i>CRCReady</i> bit can not be written from external but shows internal states.
w	write only	Reading these register bits returns always ZERO.
RFU	-	These registers are reserved for future use and shall not be changed. In case of a write access, it is recommended to write always the value "0".
RFT	-	These register bits are reserved for future use or production test and shall not be changed.

Table 9: CommandReg register (address 01h); reset value: 20h

Bit	7	6	5	4	3	2	1	0
Symbol	-		RcvOff	Power Down			Command	
Access Rights		RFU	r/w	dy			dy	

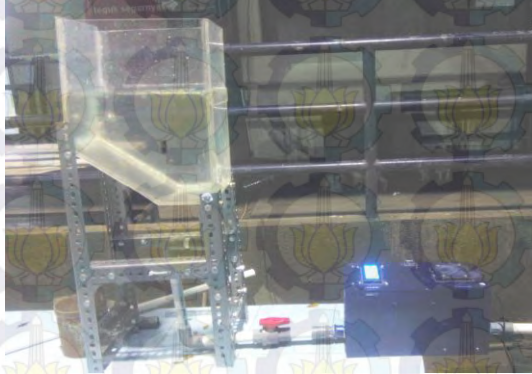
Table 10: Description of CommandReg bits

Bit	Symbol	Description
7 to 6	-	Reserved for future use.
5	RcvOff	Set to logic 1, the analog part of the receiver is switched off.
4	PowerDown	Set to logic 1, Soft Power-down mode is entered. Set to logic 0, the MFRC522 starts the wake up procedure. During this procedure this bit still shows a logic 1. A logic 0 indicates that the MFRC522 is ready for operations; see Section 16.2 "Soft Power-down" . Remark: The bit PowerDown cannot be set, when the command SoftReset has been activated.
3 to 0	Command	Activates a command according to the Command Code. Reading this register shows which command is actually executed (see Section 18.3 "MFRC522 Commands Overview").



LAMPIRAN B

Dokumentasi Alat Sistem Penghitungan Debit Air dan Pembayaran Menggunakan *Smart Card* pada PDAM



Alat Sistem Penghitungan Debit Air dan Pembayaran Menggunakan *Smart Card* pada PDAM



Tampilan penyimpanan rangkaian untuk alat Sistem Penghitungan Debit Air dan Pembayaran Menggunakan *Smart Card* pada PDAM



DAFTAR RIWAYAT HIDUP



Data Pribadi

Nama : Nur Atiqah Rianty Sari
TTL : Surabaya, 15 September 1995
Jenis Kelamin : Perempuan
Agama : Islam
Alamat : JL. Kebraon 2/124 RT 005
RW 001 Karang Pilang
Surabaya
Kode Pos : 60222
Nomor Telpn : +62 8570606 8701 (mobile)
E-mail : rianty13@mhs.ee.its.ac.id

Riwayat Pendidikan

- SDN Kebraon 2 Surabaya (2001-2007)
- SMP Negeri 24 Surabaya (2007-2010)
- SMA Negeri 10 Surabaya (2010-2013)
- D3 Teknik Elektro, ITS Surabaya (2013-Sekarang)

DAFTAR RIWAYAT HIDUP



Data Pribadi

Nama : Rifqi Robuza Rohman
TTL : Jombang, 30 April 1995
Jenis Kelamin : Laki-laki
Agama : Islam
Alamat : Jalan Halmahera VII Blok
A-03 Jombang
Kode Pos : 61419
Nomor Telpn : +62 82338802300
E-mail : rifqirobuza@gmail.com

Riwayat Pendidikan

- MIN Kauman Utara Jombang (2001-2007)
- SMP Negeri 2 Jombang (2007-2010)
- SMA Negeri 2 Jombang (2010-2013)
- D3 Teknik Elektro, ITS Surabaya (2013-Sekarang)